НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
ІМЕНІ ІГОРЯ СІКОРСЬКОГО”
Теплоенергетичний факультет
Кафедра автоматизації теплоенергетичних процесів

«На правах рукопису»
УДК 681.5.015

«До захисту допущено»
в.о. Завідувача кафедри

_____ / *В.А.Волощук*/
“____” _____ 2019 р.

Магістерська дисертація

на здобуття ступеня магістра

зі спеціальності (спеціалізації) ***151“Автоматизація та комп’ютерно-інтегровані технології” (“Комп’ютерно-інтегровані технологічні процеси та виробництва”)***

на тему: *Автоматизована система моніторингу стану пароводяного тракту прямоточного котла*

Виконав: студент II курсу, групи ТО-81мп
Якимчук Олег Анатолійович _____
(прізвище ім’я, по батькові) (підпис)

Науковий керівник *ст. викладач Любицький Сергій Вікторович* _____
(посада, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

Рецензент *Системний архітектор Степаненко О.Є.* _____
(посада, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

Засвідчую, що у цій магістерській дисертації немає запозичень з праць інших авторів без відповідних посилань.
Студент_____

Київ – 2019 року

Національний технічний університет України
“Київський політехнічний інститут
імені Ігоря Сікорського”

Факультет ***Теплоенергетичний***
Кафедра ***Автоматизації теплоенергетичних процесів***
Рівень вищої освіти – другий(магістерський) за освітньо-професійною програмою

Спеціальність (спеціалізація) *151“Автоматизація та комп’ютерно-інтегровані технології” (“Комп’ютерно-інтегровані технологічні процеси та виробництва”)*

ЗАТВЕРДЖУЮ

в.о. Завідувача кафедри

_____ /*В.А.Волощук*/
(підпис) (ініціали, прізвище)

“ “ _____ 2019 р.

ЗАВДАННЯ

на магістерську дисертацію студенту

Якимчук Олег Анатолійович
(прізвище, ім'я, по-батькові)

1. Тема дисертації Автоматизована система моніторингу пароводяного тракту прямоточного котла

науковий керівник дисертації Любицький Сергій Вікторович, ст. викладач
(прізвище, ім'я, по-батькові, науковий ступінь, вчене звання)

затверджені наказом по університету від « 4 » листопада 2019 року № 3812-с

2. Термін подання студентом дисертації «10» грудня 2019 р.

3. Об’єкт дослідження Пароводяний тракт прямоточного котла ТПП-210а

4. Предмет дослідження
Побудова статичних моделей пароводяного тракту прямоточного котла в режимі

реального часу

5. Перелік завдань, які потрібно розробити

Дослідження методів поточної ідентифікації статичних моделей об’єкту;

програмно-технічні рішення з реалізацією алгоритмів поточної ідентифікації та оцінки стану обладнання;

розробка програмного забезпечення локально рівня системи автоматизації;

розробка програмного забезпечення супервізорного рівня.

6. Орієнтований перелік графічного (ілюстративного) матеріалу

Схема автоматизації функціональна; комп’ютерна презентація

7. Орієнтований перелік публікацій

Якимчук О.А.,Любицький С.В. Реалізація SCADA-систем на основі веб-технологій // Матеріали XVII Міжнародної науково-практичної конференції аспірантів, магістрантів і студентів. – 2019. – №17. – С. 32.

Якимчук О.А.,Любицький С.В. Використання хмарних технологій в системах автоматичного керування // Матеріали XVI Міжнародної науково-практичної конференції аспірантів, магістрантів і студентів. – 2018. – №16. – С. 31.

8. Дата видачі завдання " 04 " вересня 2019 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів виконання магістерської дисертації	Строк виконання етапів магістерської дисертації	Примітка
1	*Вивчення та аналіз об'єкту автоматизації*	28.10.2019	
2	*Аналітичний огляд проблематики роботи*	02.11.2019	
3	*Опис технологічного об'єкту, його технічних характеристик та системи управління*	04.11.2019	
4	*Розробка функціональної схеми автоматизації та замовної специфікації до неї*	08.11.2019	
5	*Розробка програмного забезпечення локального рівня*	12.11.2019	
6	*Розробка програмного забезпечення супервізорного рівня*	18.11.2019	
7	*Дослідження методів поточної ідентифікації статичних моделей об'єкту*	20.11.2019	
8	*Розробка веб-додатку для моніторингу стану пароводяного тракту котла*	22.11.2019	
9	*Розробка стартап-проекту*	02.12.2019	
10	*Оформлення пояснювальної записки*	06.12.2019	
11	*Попередній захист магістерської дисертації*	10.12.2019	
12	*Захист*	18.12.2019	

Студент _____ Якимчук О.А.
(підпис) (прізвище та ініціали)

Науковий керівник дисертації _____ Любицький С.В.
(підпис) (прізвище та ініціали)

РЕФЕРАТ

Магістерську дисертації виконано на тему «Автоматизована система моніторингу стану пароводяного тракту прямоточного котла». Вона складається з пояснювальної записки, 2-ох аркушів креслень (функціональна схема автоматизації та замовна специфікація до неї), 31-го малюнка, 12-ти таблиць та комп'ютерної презентації.

Головною метою магістерської дисертації є розробка системи автоматизації пароводяного тракту прямоточного котла і веб-додатку для моніторингу стану його обладнання.

В ході виконання було розроблено програмне забезпечення локального рівня автоматизації, створено HMI/SCADA-систему та реалізовано веб-орієнтовану систему моніторингу на основі аналізу статичних моделей об'єкту керування.

Розробка веб-додатку базується на використанні мови програмування JavaScript та платформи виконання JavaScript-коду NodeJs. Клієнтський інтерфейс програми для можливості додаткових налаштувань розроблено у вигляді HTML сторінок з відповідними стилями та JavaScript логікою.

Також було проведено дослідження інвестиційної привабливості даної роботи, результати якого відображені в розділі стартап-проекту.

Публікації:

Якимчук О.А., Любицький С.В. Реалізація SCADA-систем на основі веб-технологій // Матеріали XVII Міжнародної науково-практичної конференції аспірантів, магістрантів і студентів. – 2019. – №17. – С. 32.

Якимчук О.А., Любицький С.В. Використання хмарних технологій в системах автоматичного керування // Матеріали XVI Міжнародної науково-практичної конференції аспірантів, магістрантів і студентів. – 2018. – №16. – С. 33.

Ключові слова: регулювання температури, система керування, JavaScript, NodeJs, OPCUA.

ABSTRACT

The master's thesis was performed on the topic "Automated system for monitoring the condition of the steam-water tract of a direct-flow boiler". It consists of an explanatory note, 2 sheets of drawings (functional scheme of automation and custom specifications), 31st drawing, 12 tables and a computer presentation.

The main purpose of the master's thesis is to develop a system for automation of the steam-water tract of a direct-flow boiler and a web application for monitoring the condition of its equipment.

In the course of implementation, local automation software was developed, an HMI / SCADA system was created and a web-based monitoring system was implemented based on the analysis of static models of the control object.

Web application development is based on JavaScript programming language and NodeJs JavaScript execution platform. The application's client interface is designed as HTML pages with appropriate styles and JavaScript logic for advanced settings.

A study of the investment attractiveness of this work was also conducted, the results of which are reflected in the startup project section.

Publications:

Yakymchuk O.A., Lyubitsky S.V Implementation of SCADA-systems on the basis of web technologies // Proceedings of the XVIII International scientific-practical conference of graduate students, undergraduates and students. - 2019. - №17. - P. 32.

Yakymchuk O.A., Lyubitsky S.V The use of cloud technologies in automatic control systems // Proceedings of the 16th International Scientific and Practical Conference of Graduate Students, Undergraduates and Students. - 2018. - №16. - P. 33.

Keywords: temperature control, control system, JavaScript, NodeJs, OPCUA.

ЗМІСТ

ВСТУП

Сучасний стан промислового виробництва характеризується все більш обширним впровадженням автоматизованих систем керування технологічними процесами. З метою мінімізації майбутніх фінансових витрат, а також оптимізації часу, необхідного на виготовлення кінцевого продукту, власники технологічних об’єктів все частіше звертаються до спеціалістів по впровадженню у виробництво таких рішень. Це дозволяє не тільки підвищити надійність та безпеку на даному підприємстві, але й зменшити людський фактор на кожному з етапів виробництва, переклавши відповідальність за якість виконання окремих трудомістких задач на автоматику.

Будь-який виробничий процес, не зважаючи на властивий йому рівень автоматизації, завжди залежить від надійної та безперебійної роботи обладнання, за допомогою якого він реалізовується. Як наслідок, своєчасне технічне обслуговування наявного на підприємстві устаткування є невід’ємною частиною всього циклу виробництва.

Впровадження на одному рівні з автоматизованими системами керування технологічним об’єктом додаткових систем моніторингу стану наявного там обладнання, дозволяє не тільки підвищити безпекову складову всього виробничого процесу, а й отримати значні фінансові переваги для бізнесу.

Актуальність подібного роду систем пояснюється бажанням мінімізувати час простою виробництва в ході проведення різнопланових заходів по технічному обслуговуванню устаткування.

Виявлення проблемних точок на об’єкті і попередження наслідків, до яких вони можуть привести в майбутньому, є основною метою даної роботи.

Головним завданням в ході роботи над проектом є розробка програмного додатку, здатного проводити моніторинг стану пароводяного тракту прямоточного котла в режимі реального часу.

1 АНАЛІТИЧНИЙ ОГЛЯД ПРОБЛЕМИ

1.1 Сучасний стан галузі

Розробка автоматизованих систем моніторингу стану технологічного обладнання промислових об’єктів є дуже перспективною й потрібною в сучасних реаліях розвитку автоматизації в цілому. Особливої уваги заслуговує енергетичний сектор, адже з кожним роком запит населення на електроенергію невпинно зростає.

Сучасні системи автоматизованого керування підприємствами енергетичного спрямування існують уже не один рік, та говорити про цілковите охоплення й вирішення проблем, з якими стикаються виробники, ще зарано.

Окрім того, чимала кількість українських ТЕС досі потребує капітального оновлення своїх виробничих потужностей і впровадження систем автоматизованого керування технологічними процесами. Це, звісно, має бути першочерговим завданням, що стоїть перед керівництвом таких підприємств.

Якісний підхід до вирішення цих проблем, включення до структури впровадженого на підприємстві автоматизованого комплекту додаткових систем моніторингу, дозволить забезпечити стабільний розвиток галузі і оптимізацію затрат на виробництво кінцевого продукту.

1.2 Загальна постановка задачі

З метою ефективного та високотехнологічного управління роботою пароводяного тракту прямоточного котла, в рамках даного проекту необхідно спроектувати систему автоматичного регулювання його основних технологічних параметрів на базі сучасної контроллерної техніки та автоматики.

Реалізація супервізорного рівня контролю за технологічними параметрами об’єкту керування (SCADA-системи) має базуватись на основі використання технології OPC, а також забезпечувати можливість архівації даних, побудови трендів, налаштувань алармів тощо.

При розробці автоматизованої системи моніторингу стану обладнання, потрібно передбачити можливість швидкого розширення функціоналу такого додатку, впровадження додаткових алгоритмів побудови статичних моделей, забезпечити зручну взаємодію користувача з представленим функціоналом і можливість передачі отриманих результатів назад до SCADA-системи.

Окрім того, при виборі технологій розробки слід врахувати останні тенденції в сфері програмування й принципи веб-орієнтованої роботи майже всіх новинок на ринку.

2 ОПИС ОБ'ЄКТУ УПРАВЛІННЯ

2.1 Котельний агрегат ТПП-210А

Даний вид котельного агрегату типу ТПП-210А є прямоточним, володіє потужністю 300 МВт, виготовлений на таганрогському котельному заводі. Тип компоновки - П-образна, в двухкорпусному виконанні призначений для спалювання вугілля марки АШ з рідким шлаковидаленням [2].

Також даний вид котлів може бути використаний для спалювання природного газу, тому його пальники виконані пило-газовими. В якості ропалювальної речовини виступає мазут [3]. Для підсвічування топки при нестійкому горінні вугілля марки АШ використовується мазут та природний газ. Схема пило підготовки індивідуальна із загальним для обох корпусів пиловим бункером і трьома пилосистемами. Кожна пилосистема замкнутого типу з шаробарабанними млинами типу ШБМ370-850. Топка котла обладнана шістьома пилогазовими пальниками продуктивністю 11,2 т/год, розташованими зустрічно в один ярус на фронтовій і задній стінках [4].

Пароводяний тракт котла виконаний 4-х поточним з самостійним регулюванням витрати і температури пари по кожному потоці (на один корпус передбачено два потоки). Регулювання температури первинної пари здійснюється вприскуванням живильної води в пароохолоджувачі.

Котлоагрегат ТПП-210А обладнаний: двома відцентровими дуттєвими вентиляторами двостороннього всмоктування типу ВДН-24х2; двома регенеративними повітронагрівачами типу РВВ-68МА; двома 2-х швидкісними димососами типу ДО-31,5 [1].

Парова турбіна типу К-300-240 ХТГЗ конденсаційна, одновальна, трициліндрова з регенеративними відборами пари і проміжним паропеперегрівом. Пароросподіл турбіни-сопловий з двома блоками. У кожному блоці розміщено по одному стопорному і по три регулюючих клапана.

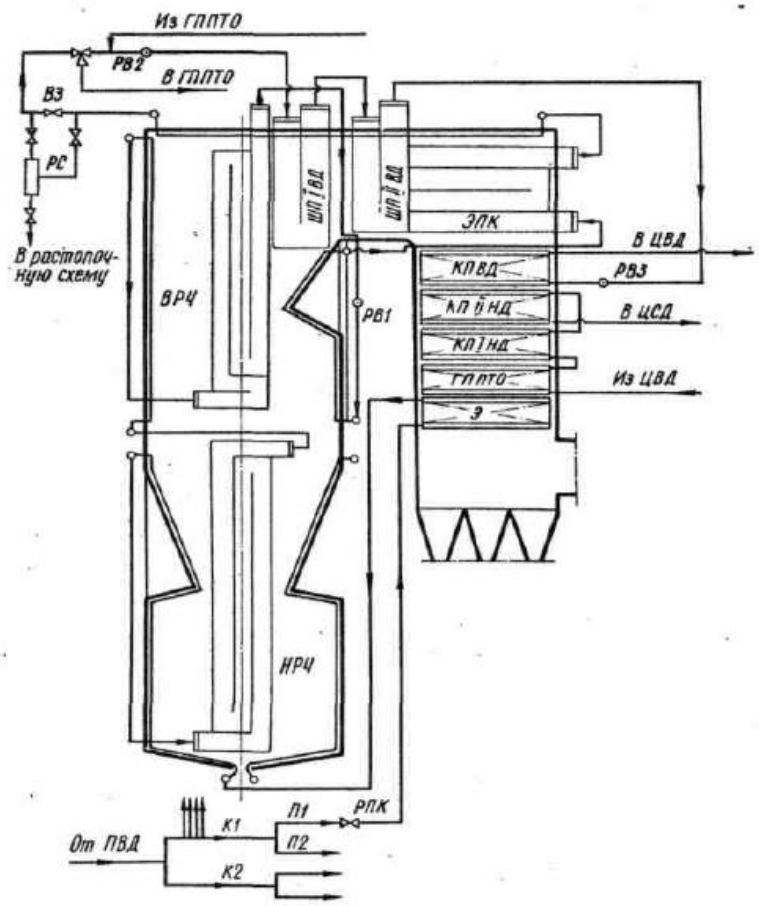
Живильна установка складається з одного основного живильного турбонасоса типу ОСПТ-1150м і одного пускорезервного електронасоса типу ПЕ-600-300 з гідромуфтою [4].

Швидкодіюча редукційна охолоджувальна установка продуктивністю 150 тонн/год забезпечує скидання навантаження і регулює пускові режими енергетичного блоку.

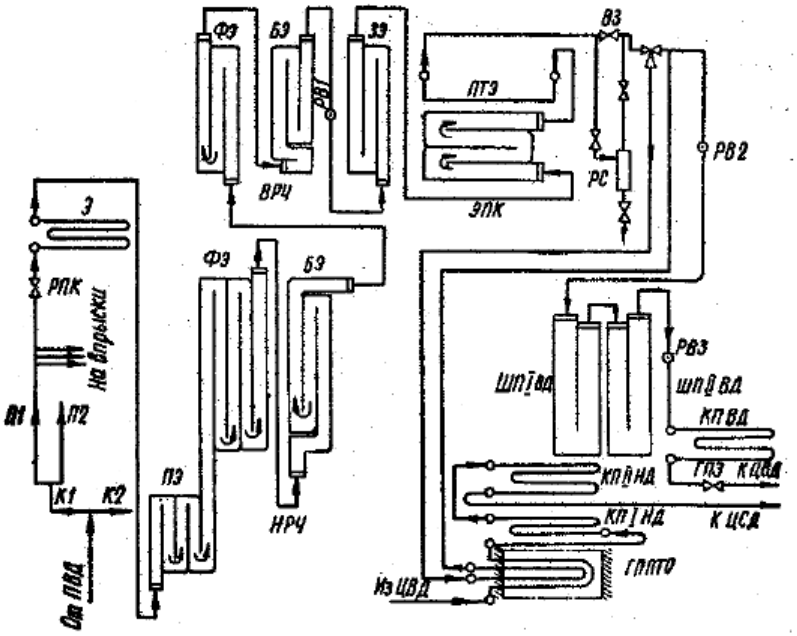
Регульований діапазон навантажень енергетичного облоку за умови відності змін у скліді обладнання і горілчаного апарату сягає 300-225 МВт при роботі на пиловугільному паливі і 300-180 мВт при роботі з використанням природного газу.

Експлуатація пиловугільного блоку проводиться в режимах комбінованого тиску пари предед турбіною. За умови дотримання змін поточного навантаження в діапазоні від 300 до 250МВт, номінальний тиск пару становитиме 240 кгс/см2. Це досягається за рахунок дії на регулюючі клапани турбіни аж до цілковитого їх закриття. У випадку зниження навантеження на енегетичний блок до позначки 225МВт, відбувається перехід на так званий ковзкий тиск з повністю відкритими регулювальними клапанами турбіни [3].

Мінімальний значення тиск пару перед турбіною має значення 167 кгс/см^2.



Мал. 2.1 Повздовжній розріз пароводяного котла ТПП-210



Мал. 2.2. Схема пароводяного тракту котла ТПП-210

2.2 Технічні характеристики об’єкту управління

Паровий котел ТПП-210А має ряд наступних технічних характеристик, що відображено таблиці 1.1.

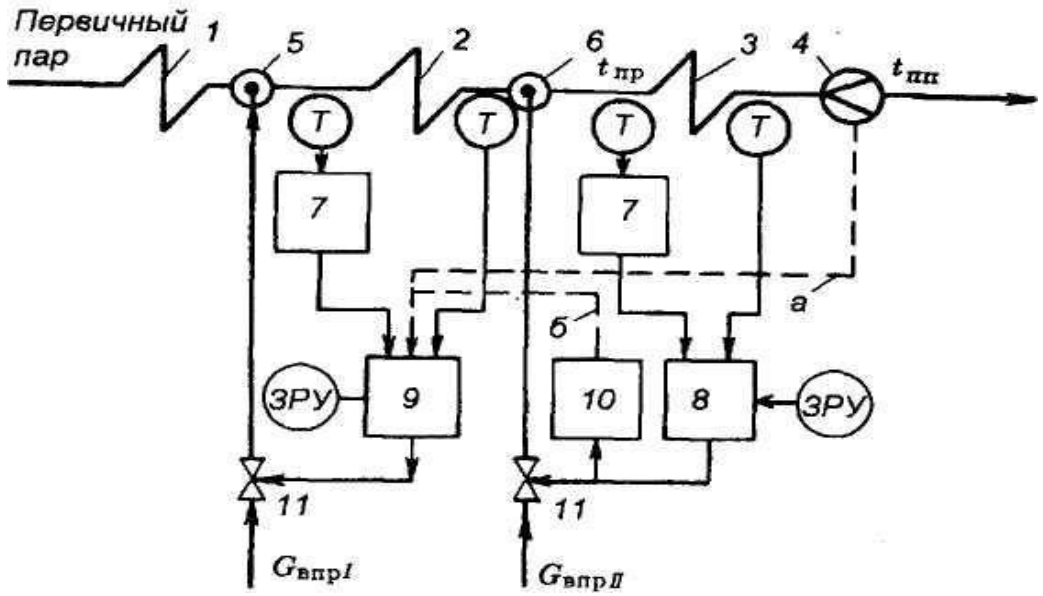
Таблиця 1.1. Нормативні значення технологічний параметрів

Параметр	**Значення**
Паропродуктивність	950 т/год
Температура живильної води	272 °С
Температура первинного пару	545 °С
Температура вторинного пару	545 °С
Витрата природного газу	300 $м^3$/ год
Тиск вторинного пару	39,5 $кгс/см^2$
Тиск первинного пару	255 $кгс/см^2$
Витрата вторинного пару	240 т/год
Витрата первинного пару	230 т/год

2.3 Огляд системи управління температурою перегрітого пару

Регулювання температури пари послідовно включених перегріваючих ділянок первинного тракту здійснюють за допомогою АСР вприсків, що працюють по двухімпульсній схемі [3].

Схема автоматичної системи регулювання температури первинного пара одного з циркуляційних контурів прямоточного котла з двома вприсками зображена на мал. 2.3.



Мал. 2.3. Схема автоматичної системи регулювання температури первинної пари

Регулятори (8,9) підтримують задану температуру первинного пара ($t_{пп}$) шляхом впливу на регулюючі клапани шиберного типу (11), що змінюють витрату води на пароохолоджувачі (5,6).

Автоматичний регулятор температури пара отримує основний сигнал по температурі за ступенями пароперегрівачів. Основний сигнал порівнюється з сигналом від задатчика і забезпечує необхідну точність регулювання. По цьому каналу регулятор реалізує ПІ - закон регулювання.

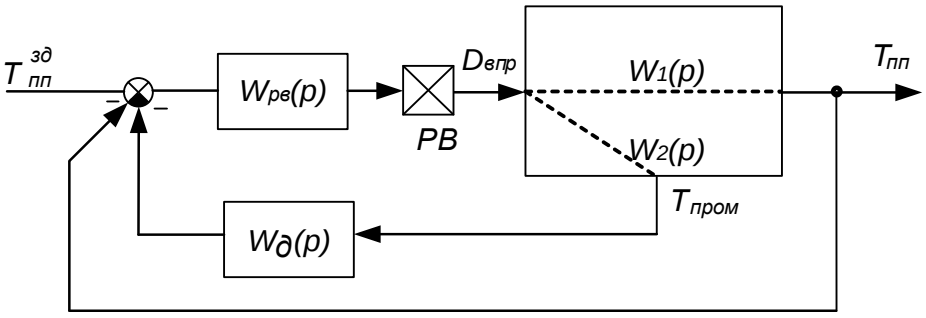
Зважаючи на значну інерційності пароперегрівача при зміні витрати води на вприскування, передбачений додатковий випереджаючий (швидкісний) сигнал за вприскувачами. Даний сигнал по температурі пара подається на регулятор через диференціатор (7), який виробляє сигнал, приблизно пропорційний швидкості зміни температури за пароохолоджувачем.

При наявності цього сигналу регулятор почне змінювати витрату води на вприскування в пароперегрівач, коли температура пара на його виході ще не

змінилася. Введення цього додаткового сигналу значно покращує якість регулювання температури пари.

У випадку використання звичайної одноконтурної системи реакція на зміни температури пари буду більш тривалою, що в свою чергу значно погіршить показники якості роботи системи в цілому.

Застосування диференціатора також необхідно для усунення залишкової нерівномірності, що вводиться сигналом від додаткової термопари, встановленої за пароохолоджувачем. З огляду на застосування диференціатора, додатковий сигнал від термопари, встановленої за пароохолоджувачем, подається на регулятор тільки в перехідних режимах, пов'язаних зі зміною витрати води на вприскування. Відповідна розрахункова структура АСР зображена на мал. 2.4



Мал. 2.4. Розрахункова структура АСР с дифференціатором температури перегрітого пара

3 РОЗРОБКА СИСТЕМИ УПРАВЛІННЯ ОБ'ЄКТОМ

Система управління технологічним об'єктом керування в рамках даної роботи складається з трьох основних частини: рівня супервізорного контролю (Scada-система), нижнього рівня керування або ж локального (контролер з вимірювальною та виконавчою апаратурою) та веб-додатку, що здійснює моніторинг технологічного стану виробничого обладнання.

В якості супервізорного рівня застосовується програмний продукт SimpleScada, призначений для створення сучасних людино-машинних інтерфейсів.

Нижній рівень розробленої системи побудований на базі надійних промислових контролерів фірми Siemens – Simatic S7-400. Цей виробник уже довгий час є світовим лідером в сфері автоматизації і пропонує широкий спектр рішень для систем автоматичного керування.

Розробка веб-додатку базуватиметься на основі використання мови програмування JavaScript та платформи виконання JavaScript-коду NodeJs. Веб-орієнтованість даних технологій і вільний доступ їх застосування, дозволить написати добре розширюваний функціонал, що буде здатний швидко й якісно виконувати поставленні завдання.

На нижньому рівні автоматизованої системи здійснюється неперервне регулювання і програмно-логічне управління технологічним процесом. Тут виконуються функції контролю, регулювання, захисту та блокування.

В свою чергу верхній рівень (супервізорний), являє собою HMI/SCADA–систему, що повинна реалізовувати набір стандартних функції:

- обмін даними з контролерами;

- архівування даних;
- візуалізація технологічного процесу у вигляді мнемосхем;
- ведення історичних алармів і алармів реального часу;
- побудова історичних трендів і трендів реального часу;

3.1 Опис програмного забезпечення локального рівня

Програмне забезпечення локального рівня було створене за допомогою середовища програмування контролерів TIA Portal V15.

Виконувана контролером програма забезпечує регулювання технологічних параметрів пароводяного тракту котла. Її основним призначенням є обрахунок потрібної в даний момент керуючої дії, що здійснюється на основі вхідної інформації і значень додаткового інформаційного сигналу.

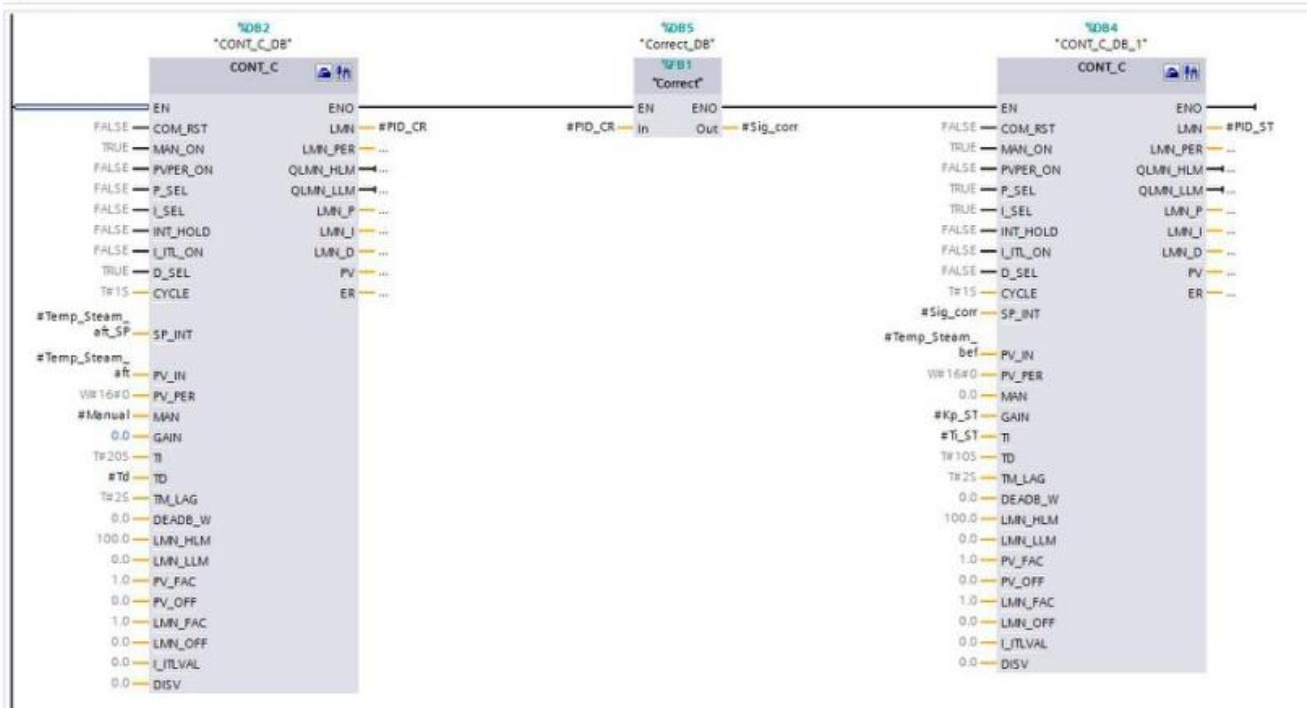
Реалізація процесу регулювання зумовлює необхідність вводу на модуль АЦП контролера сигналів від давачів температури. Вони далі опрацьовується програмою контролера і на основі отриманих результатів програма формує регулюючий вплив на виконавчий механізм. Список змінних, які було використано в програмі наведено на мал. 3.1.

Data_block_1

	Name	Data type	Offset	Start value	Retain	Visible in ...	Setpoint
1	Static						
2	Kp_ST	Real	0.0	0.0	☑	☑	☐
3	Ti_ST	Time	4.0	T#0ms	☑	☑	☐
4	Td	Time	8.0	T#0ms	☑	☑	☐
5	Temp_Steam_aft_SP	Real	12.0	0.0	☑	☑	☐
6	Temp_Steam_bef	Real	16.0	0.0	☑	☑	☐
7	Temp_Steam_aft	Real	20.0	0.0	☑	☑	☐
8	Manual	Real	24.0	0.0	☑	☑	☐
9	Actuator	Real	28.0	0.0	☑	☑	☐
10	Preasure_PSteam	Real	32.0	0.0	☑	☑	☐
11	Flow_PSteam	Real	36.0	0.0	☑	☑	☐
12	HiTemp_Steam_aft	Real	40.0	0.0	☑	☑	☐
13	LoTemp_Steam_aft	Real	44.0	0.0	☑	☑	☐
14	HiPreasure_PSteam	Real	48.0	0.0	☑	☑	☐
15	LoPreasure_PSteam	Real	52.0	0.0	☑	☑	☐
16	HiTemp_PSteam	Real	56.0	0.0	☑	☑	☐
17	LoTemp_PSteam	Real	60.0	0.0	☑	☑	☐
18	HiFlow_PSteam	Real	64.0	0.0	☑	☑	☐
19	LoFlow_PSteam	Real	68.0	0.0	☑	☑	☐
20	Temp_Steam_aftAlarm	Bool	72.0	false	☑	☑	☐
21	Preasure_PSteam Alarm	Bool	72.1	false	☑	☑	☐
22	Temp_PSteamAlarm	Bool	72.2	false	☑	☑	☐
23	Flow_PSteamAlarm	Bool	72.3	false	☑	☑	☐
24	Temp_PSteam	Real	74.0	0.0	☑	☑	☐

Мал. 3.1. Список використаних в програмі змінних

В середовищі програмування TiaPortal було реалізовано роботу контуру регулювання температури за ширмовим пароперегрівачем першого ступеня. З цією метою було розроблено функціональний блок, який відтворює принцип двухімпульсного керування (мал. 3.2).



Мал. 3.2. Функціональний блок, що реалізовує двухімпульсне регулювання температури пари

Сигнал від диференціатора поступає на вхід ПІ-регулятора , що в свою чергу здійснює керування роботою клапана води, призначеного для впирску.

Температура пару за ширмовим пароперегрівачем поступає на контролер в якості основної регулюючої величини (інерційного сигналу), а значення температури пару перед ним – в якості швидкісного інформаційного сигналу. Такий механізм дозволяє значною мірою пришвидшити реакцію системи на виникаючі збурення.

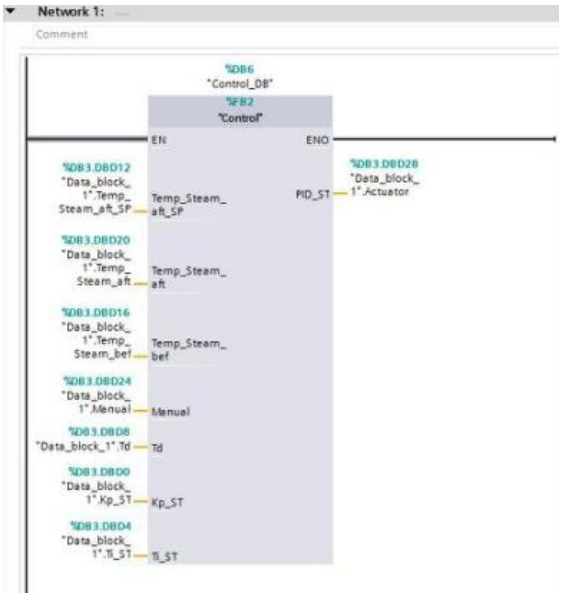
Окрім того, в розглянутому блоці передбачено можливість зміни режиму роботи регулятора з автоматичного на ручний та навпаки. За цей процес відповідає змінна «Manual».

Обов’язковою складовою системи керування є розробка окремих функціональних блоків, призначених для виявлення аварійних ситуацій у разі відхилень технологічного параметру за допустимі для нього межі. На мал. 3.3 наведено один з таких блоків.

Мал. 3.3. Функціональний блок для виявлення аварійної ситуації

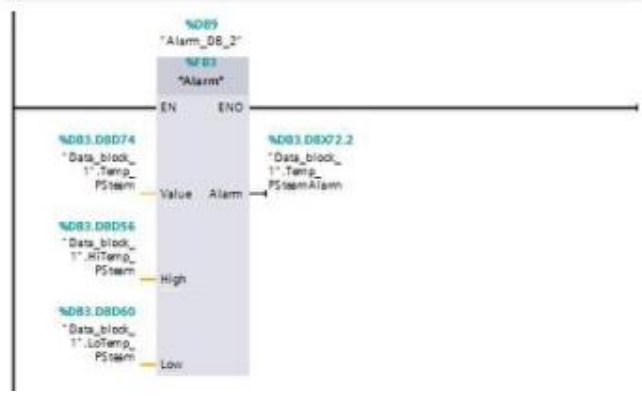
Представлений алгоритм реалізовує своєчасну зміну значення змінної #Alarm (призначеної для індикації спрацювання сигналізації) в TRUE (FALSE) у випадку виходу вхідного параметру за допустимі межі.

Загальний вигляд функціонального блоку, призначеного для регулювання температури перегрітого пари, а також масив його вхідних параметрів можна побачити на мал. 3.4.

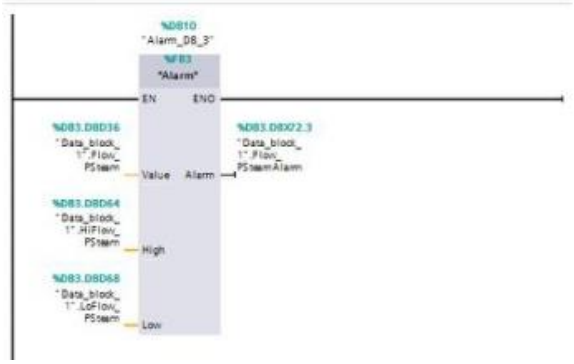


Мал. 3.4. Функціональний блок для регулювання температури перегрітого пару

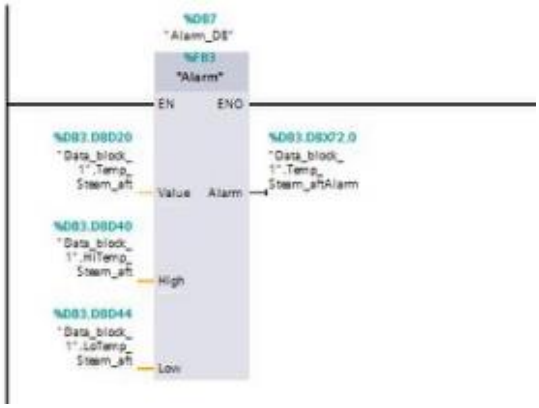
Малюнки 3.5 – 3.7 демонструють інші функціональні блоки, призначені для фіксування аварійних ситуацій в системі.



Мал. 3.5. Функціональний блок для фіксації аварій пов’язаних з температурою первинного пару



Мал. 3.6. Функціональний блок для фіксації аварій пов’язаних з витратою первинного пару



Мал. 3.7 Функціональний блок для фіксації аварій пов’язаних з температурою перегрітої пари за ширмовим пароперегрівником першого ступеня

3.2 Опис програмного забезпечення супервізорного рівня

Програмне забезпечення верхнього рівня системи автоматизації представлене у вигляді SCADA-системи. В рамках розробленого проектного рішення, цією системою забезпечується наступний функціонал:

• обмін даними з контролером контролера;

• архівування даних;

• візуальне представлення технологічного процесу в якості мнемосхеми;

• ведення історичних процесів і процесів, як відбуваються в режимі реального часу;

• відображення історичних трендів та трендів реального часу;

• можливість задання значень окремих технологічних параметрів.

Також було розроблено набір необхідних для роботи оператора розділів програми. Серед них такі як:

• головна сторінка з відображенням мнемосхеми;

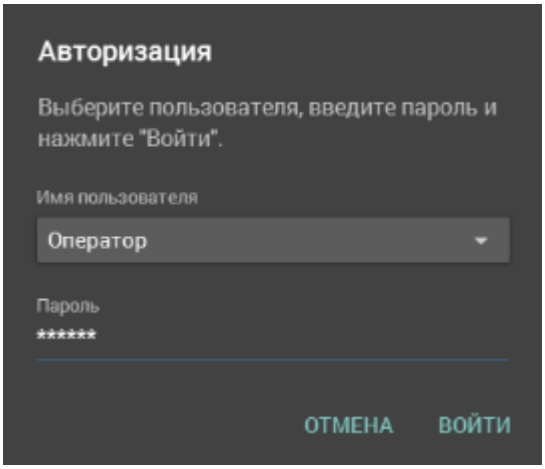
• сторінка виникаючих в ході технологічного процесу тривог;

• сторінка відображення трендів;

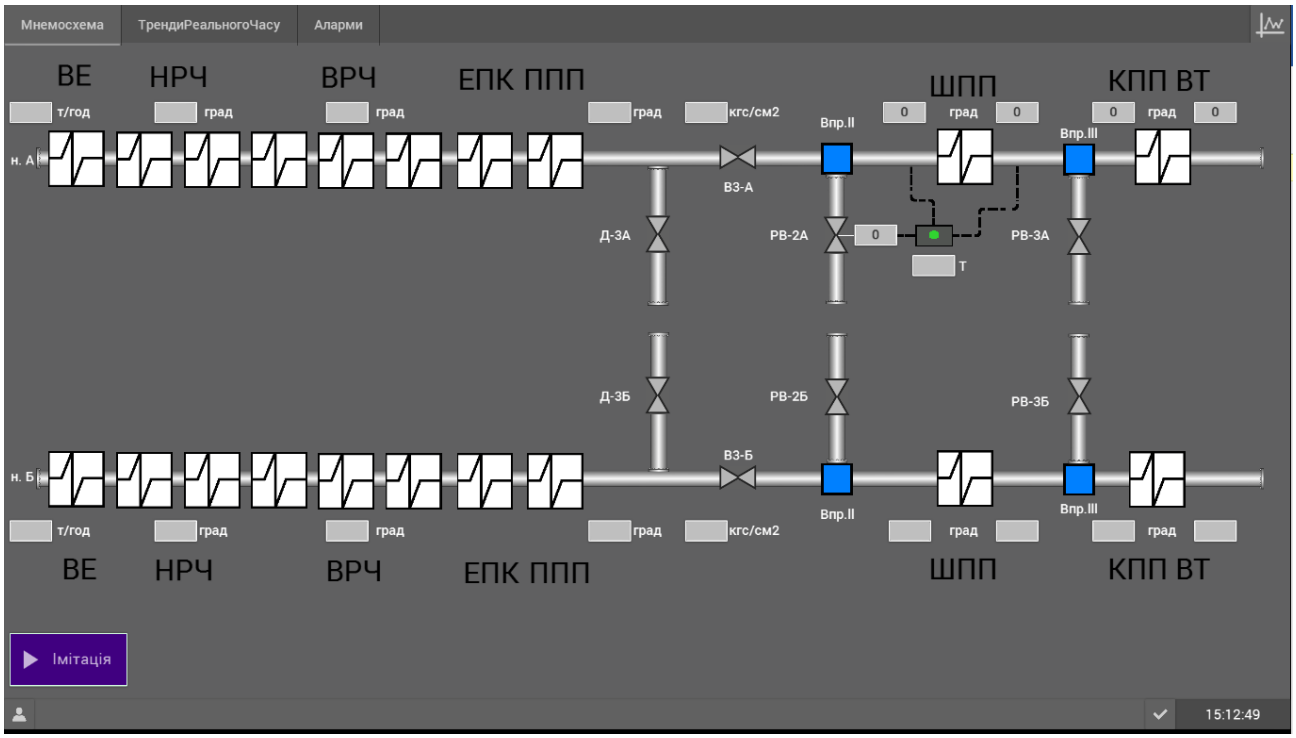
• інші додаткові вікна (наприклад форма авторизації користувача і вікно імітаційного налаштування значень технологічних параметрів).

Розпочнемо з вікна авторизації користувача, що виконує допоміжну функцію і зображено на мал.3.8. Його основне призначення – недопущення до системи керування технологічним об’єктом людей, які не мають на то права.

Сторінку з головною мнемосхемою на мал.3.9 буде відображено лише при наявності успішній авторизації. Окрім доступу до технологічних параметрів об’єкту, авторизований користувач також зможе користуватись навігаційним меню для переходу по сторінках SCADA-системи.

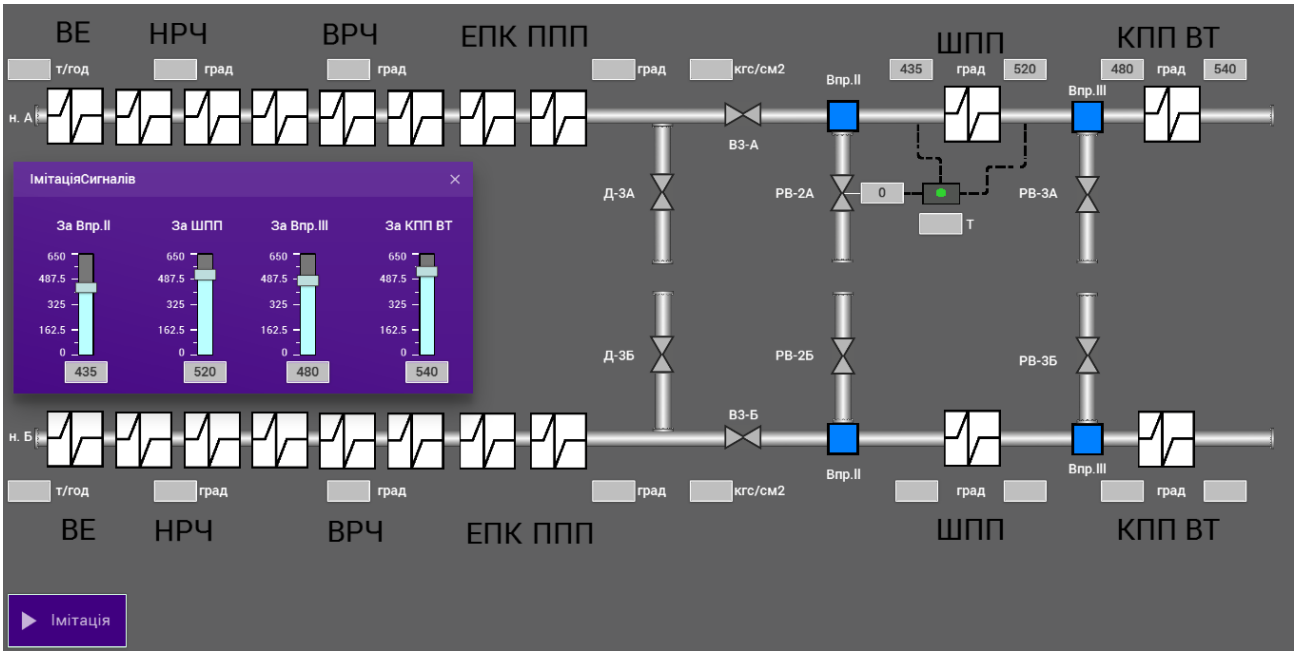


Мал. 3.8 Вікно авторизації



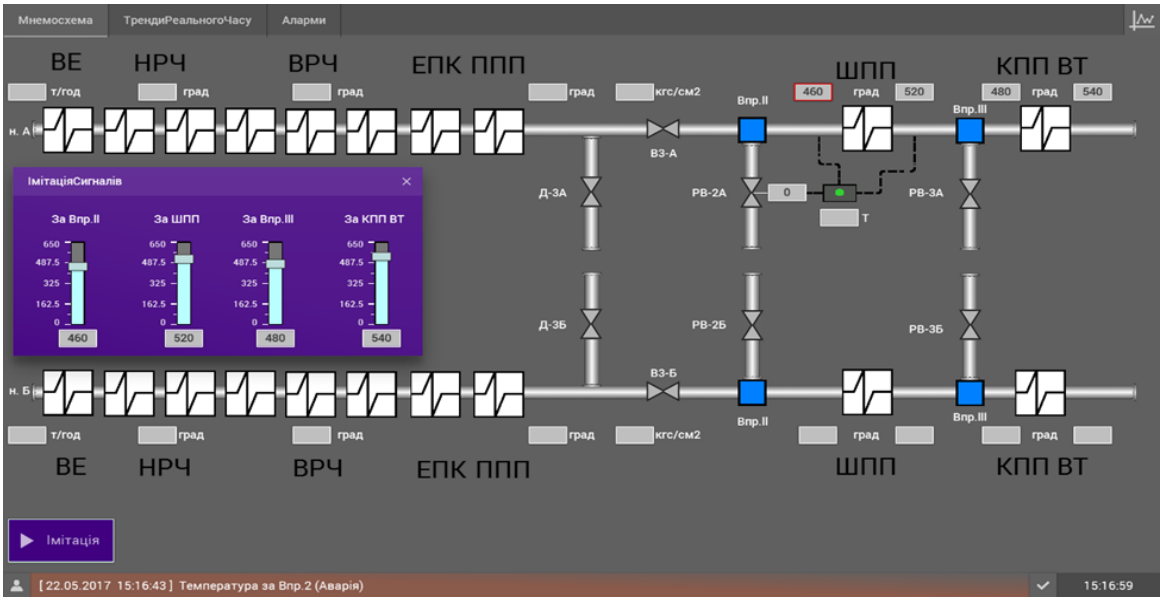
Мал. 3.9. Головна сторінка з мнемосхемою

Вікно налаштувань імітаційних значень технологічних параметрів представлено на мал. 3.10.



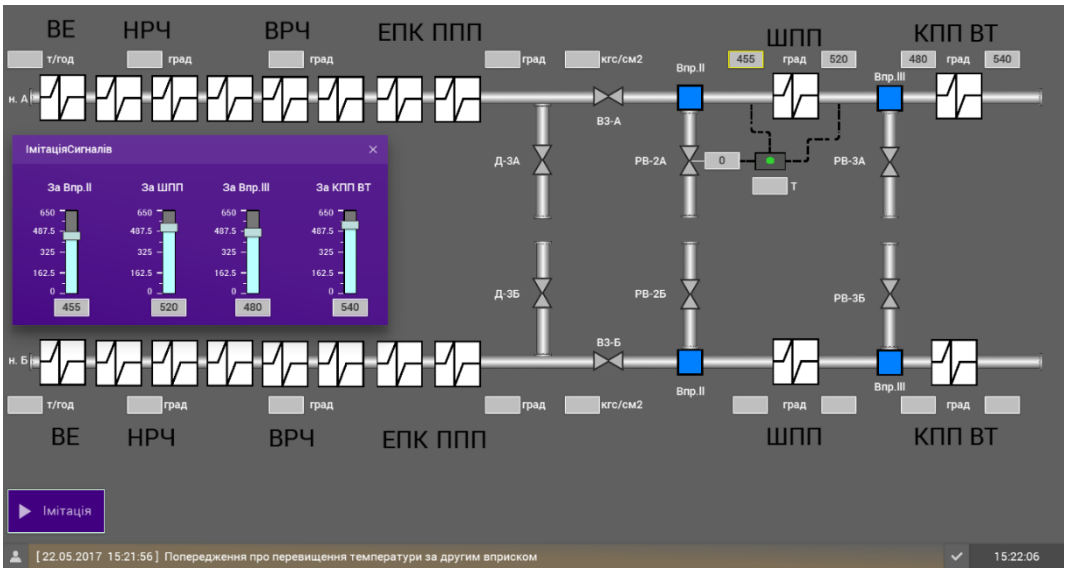
Мал. 3.10. Вікно налаштування імітаційних значень технологічних параметрів

У разі виникнення аварійних ситуацій, на мнемосхемі відбувається індикація відповідного технологічного параметру червоним кольором, а внизу сторінки виводиться короткий текстовий опис даної проблеми. Відповідний ситуація зображена на мал.3.11.



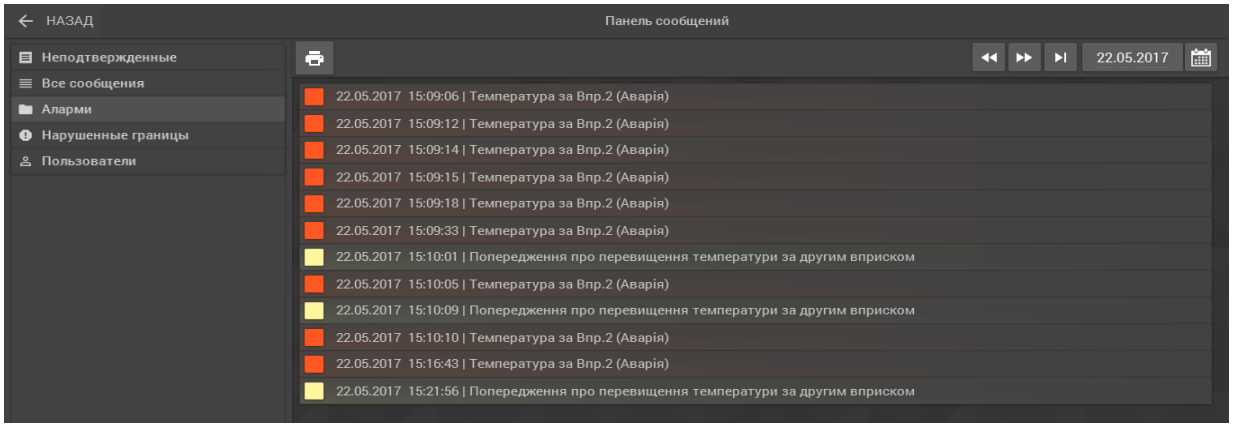
Мал.3.11. Виникнення аварійної ситуації

В налаштування кожного з технологічних параметрів можна також задати попереджувальні повідомлення, що будуть викликатись у разі виходу його значення за попередньо вказані межі. Відповідний процес зображено на мал. 3.12.

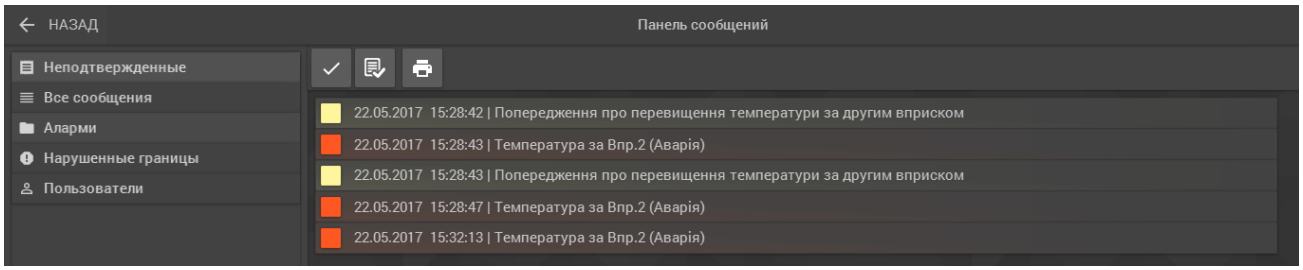


Мал. 3.12. Вивід попередження про вихід значення технологічного параметру за допустимі межі

Сторінка алармів, що зображена на мал. 3.13, відповідає за процес архівування виникаючих тривог і запису їх в базу даних. Історичні аларми зберігаються в зовнішніх СУБД, а саме база даних MySQL. Вони описують вихід усіх наявних технологічних параметрів за вказані допустимі їх межі. Оператор може підтвердити будь-яку аварію у вікні представленому на мал. 3.14.

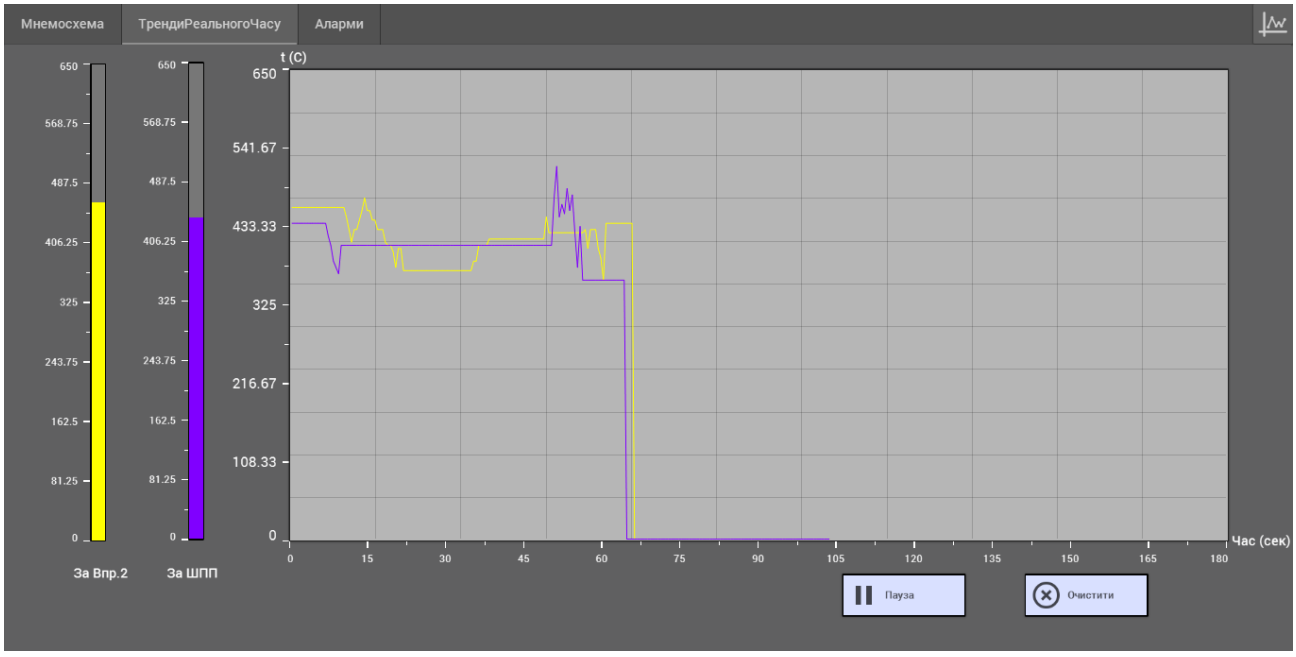


Мал. 3.13. Вікно відображення алармів



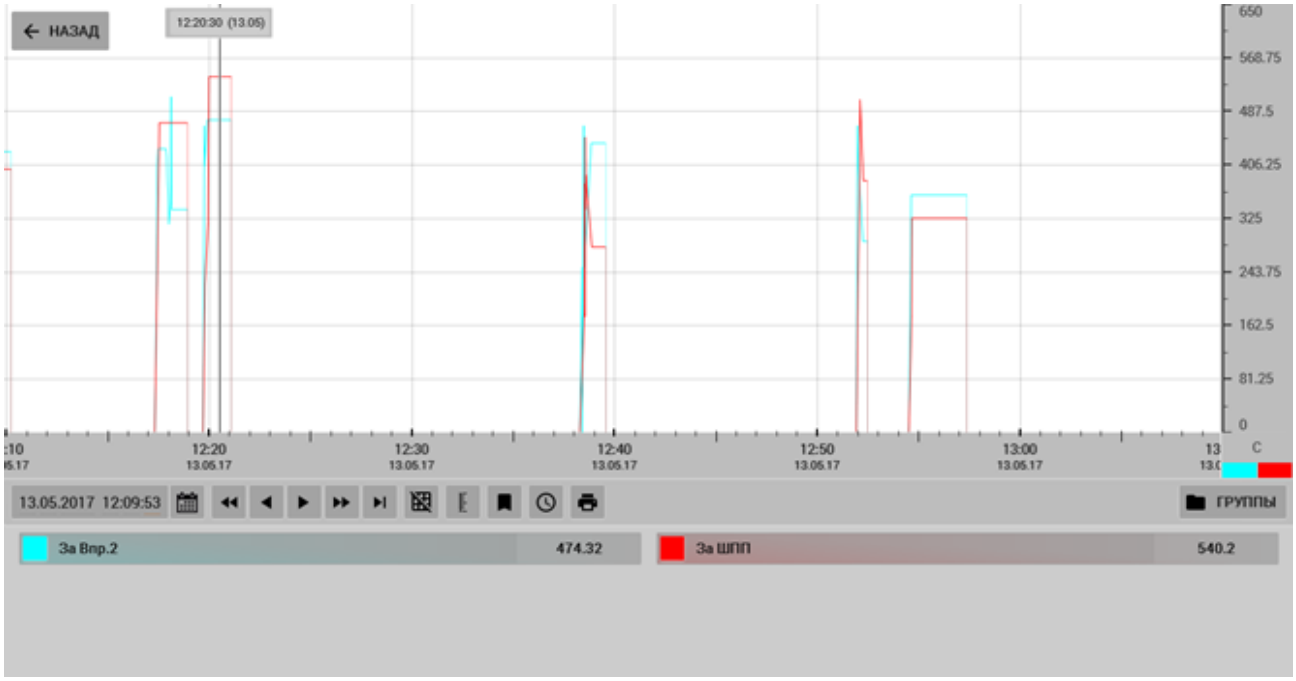
Мал. 3.14. Вікно для підтвердження виникаючих алармів

На мал.3.15 представлено сторінку з відображенням трендів реального часу, що демонструє значення регульованих параметрів з плином часу.



Мал. 3.15. Сторінка з відображенням трендів реального часу

На сторінці представленій на мал. 3.16, також є можливість перегляду історичних трендів, зчитування даних для яких відбувається з зовнішньої бази даних. Тут є можливість обрати певний часовий діапазон та відфільтрувати вивід потрібних параметрів.



Мал. 3.16. Сторінка з відображенням історичних трендів

4. РОЗРОБКА ДОДАТКУ ДЛЯ МОНІТОРИНГУ СТАНУ ПАРОВОДЯНОГО ТРАКТУ КОТЛА

Сучасні тенденції в розробці різного роду програмного забезпечення усі в тій чи іншій мірі тяжіють до веб-технологій. Кожний користувач персонального комп’ютера має досвід роботи з веб-браузерами. Тому, розробка інтерфейсу додатку проводилась саме на основі таких технологій як: HTML5, CSS3 та JavaScript.

Для створення алгоритму обрахунку статичних моделей об’єкту і реалізації процесу взаємодії серверу з користувачем найкраще підходить платформа серверного виконання JavaScript коду - Node.js. Використання тієї ж самої мови програмування на клієнті та сервері дозволяє не тільки спростити інтеграцію цих частин додатку, а й певним чином уніфікувати підбір програмістів, які апріорі повинні володіти навиками роботи з JavaScript. Це в свою чергу зробить їх більш потрібними і корисними на кожному з етапів розробки.

Усі власники виробництв для якнайкращої роботи й оптимізації своїх технологічних процесів обирають досить різноманітне як обладнання, так і програмне забезпечення, що працює з ним. Задля якісного та широкого охоплення ринку, необхідно було обрати найбільш підходящий спосіб взаємодії додатку із софтом, що уже використовується замовником. Тому повністю обґрунтованим є рішення використовувати обмін потрібними даними за технологією OPCUA, що надає єдиний інтерфейс для управління об'єктами і його технологічними процесами, а також широко використовується як у вітчизняній так і в закордонній промисловій автоматизації.

4.1 Опис технології OPC

OPC - це набір специфікацій стандартів, який було розроблено з метою скоротити витрати на створення і супровід додатків промислової автоматизації. Його суть досить проста - надати розробникам промислових програм універсальний

фіксований інтерфейс (тобто набір функцій) обміну даними з будь-якими пристроями [9].

В той же час, розробники пристроїв надають програму, що реалізовує цей інтерфейс (набір функцій). Коротку інформацію про основні й найбільш вживані специфікації, що входять до стандарту OPC, наводимо в наступних пунктах роботи.

4.1.1 OPC Data Access

Специфікація OPC DA (Data Access) описує набір функцій обміну даними між сервером та клієнтом в режимі реального часу. Найбільш широко використовувана специфікація. Для сервера визначає поняття структурного елементу, яким є технологічний параметр, поняття групи елементів і способи надання інформації для групи таких елементів.

Для клієнтської програми специфікація визначає способи запиту у сервера групи технологічних параметрів і їх значень.

4.1.2 OPC Alarms and Events

Специфікація OPC AE (Alarms & Events) створена для контролю за різними подіями, наприклад, виходом значення за якісь межі, обриву сигналу, діями операторів тощо.

Використовуючи цей стандарт, в режимі реального часу можна отримати відомості про аварійні ситуації, дії оператора або ж просто певні інформативні повідомлення. Функціонально це може бути виконано як за запитом користувача, так і в автоматичному режимі (наприклад повідомлення про виникнення аварій чи позаштатних ситуацій).

Для програми-клієнта специфікація визначає способи формування запитів про групу тих чи інших технологічних параметрів об'єкта, а також інформації щодо

виникнення пов'язаних з ними подій. Для серверної ж програми вона визначає формат відповіді на отриманий запит та алгоритм передачі потрібних даних.

4.1.3 Специфікація OPC UA

Незважаючи на величезний успіх і загальне визнання, практика виявила такі недоліки OPC технології:

- доступність тільки на операційних системах сімейства Microsoft Windows;
- зв'язок с технологією DCOM, вихідні коди якої є закритими. Це не дозволяє вирішувати питання надійності ПО, а також виявляти і усувати виникаючі програмні відмови;
- бувають проблеми конфігурації, пов'язані з DCOM;
- неточні повідомлення DCOM про переривання зв'язку;
- непристосованість DCOM для обміну даними через інтернет;
- непристосованість DCOM для забезпечення інформаційної безпеки. [9]

У зв'язку з цим в 2006 році OPC Foundation запропонував нову стандартну специфікацію для обміну даними в системах промислової автоматизації, що отримала назву "OPC Unified Architecture" - "OPC з уніфікованою архітектурою", яка розглядається як OPC стандарт нового покоління.

Стандарт OPC UA встановлює методи обміну повідомленнями між OPC сервером і клієнтом, які не залежать від апаратно-програмної платформи, від типу взаємодіючих систем і мереж.

OPC UA забезпечує надійну і безпечну комунікацію, протидію вірусним атакам, гарантує ідентичність інформації клієнта і сервера.

У новому стандарті використовується поняття об'єкта, під яким розуміється фізичний або абстрактний елемент системи. Прикладами об'єктів можуть бути фізичні пристрої включно з їх системами і підсистемами. Датчик температури, наприклад, може бути представлений як об'єкт, який включає в себе значення температури, набір параметрів сигналізації і меж її спрацьовування [9].

Об'єкт, за аналогією з об'єктно-орієнтованим програмуванням, визначається як екземпляр класу, а клас розглядається як тип даних. Об'єкти включають в себе змінні, події і методи.

OPC UA використовує кілька різних форматів даних, основними з яких є бінарні структури і XML документи. Формат даних може бути визначено постачальником OPC сервера або стандартом. Для роботи з довільними форматами клієнт може запросити у сервера інформацію про опис цього формату. У багатьох випадках використовується автоматичне визначення формату даних під час їх передачі.

OPC UA має високу робастність даних і повідомлень про події. Робастність забезпечується механізмом швидкого виявлення помилок комунікації і відновлення даних.

Сервери можуть мати доступ як до поточних, так і архівованих даних, до подій і аварійних сигналів. OPCUA може бути впроваджений в різні комунікаційні протоколи, а дані можуть бути закодовані способами, оптимальними по співвідношенню ефективності і переносимості на інші платформи.

OPCUA - це незалежний від платформи стандарт, за допомогою якого системи і пристрої різного типу можуть взаємодіяти шляхом відправки повідомлень між клієнтом і сервером через різні типи мереж. Протокол підтримує безпечну взаємодію шляхом валідації клієнтів і серверів, а також протидії атакам.

OPCUA визначає поняття сервісів, які сервери можуть надавати, а також сервіси, які сервер підтримує для клієнта. Інформація передається у вигляді типів даних, визначених OPCUA і виробником. Крім того, сервера визначають об'єктну модель, для якої клієнти можуть здійснювати динамічний огляд.

Специфікація OPCUA надає поєднання інтегрованого адресного простору і сервісної моделі. Це дозволяє серверу інтегрувати дані, повідомлення (Alarms), події (Events) і історію в цьому адресному просторі, а також надавати доступ до них за допомогою інтегрованих сервісів.

Сервіси також надають інтегровану модель безпеки. OPCUA дозволяє серверним програмам надавати клієнтам визначення типів для доступу до об'єктів з адресного простору.

Стандарт OPCUA також додає підтримку множинної зв'язності між вузлами замість простого обмеження ієрархічністю. Така гнучкість в комбінації з визначенням типів дозволяє застосовувати даний стандарт для вирішення завдань в широкій проблемній області. Інформаційна модель OPC більше не ґрунтується тільки на ієрархічних папках, елементах і властивостях. Замість цього використовується, так звана, повна коміркова мережа, заснована на вузлах. Крім того, ці вузли можуть передавати все різноманіття метаінформації.

У найпростішому випадку вузол можна порівняти з об'єктом, відомим з об'єктно-орієнтованого програмування. Він має атрибути доступні для читання, методи, які можуть бути викликані, і ініціюючі події, які можуть бути передані. Вузли використовуються для обробки даних нарівні з усіма іншими типами метаданих.

Специфікація OPCUA була створена для забезпечення надійної та безперебійної передачі даних. Основна особливість всіх OPC серверів - здатність видавати як окремі дані, так і пов’язані з ними події.

OPCUA спроектований для підтримки широкого діапазону серверів, від простих ПЛК до промислових серверів. Ці сервера характеризуються широким спектром розмірів, продуктивності, платформ виконання та функціональної ємності. Для забезпечення сумісності OPCUA визначає підмножини, іменовані профілями, які сервери можуть вказувати для узгодження. Клієнти, як наслідок, отримують можливість виконувати огляд профілів сервера і здійснювати взаємодію з ним на їх основі.

Дана специфікація спроектована як ядро в шарі, ізольованому від підлягаючих комп'ютерних технологій і мережевих транспортів. Це дозволяє їй при необхідності

розширюватися з урахуванням майбутніх технологій без особливих змін свої базової архітектури.

На даний момент специфікацією визначені два способу кодування даних: XML і UA Binary. Додатково, визначено два типи транспортного шару: TCP і HTTP/SOAP [9].

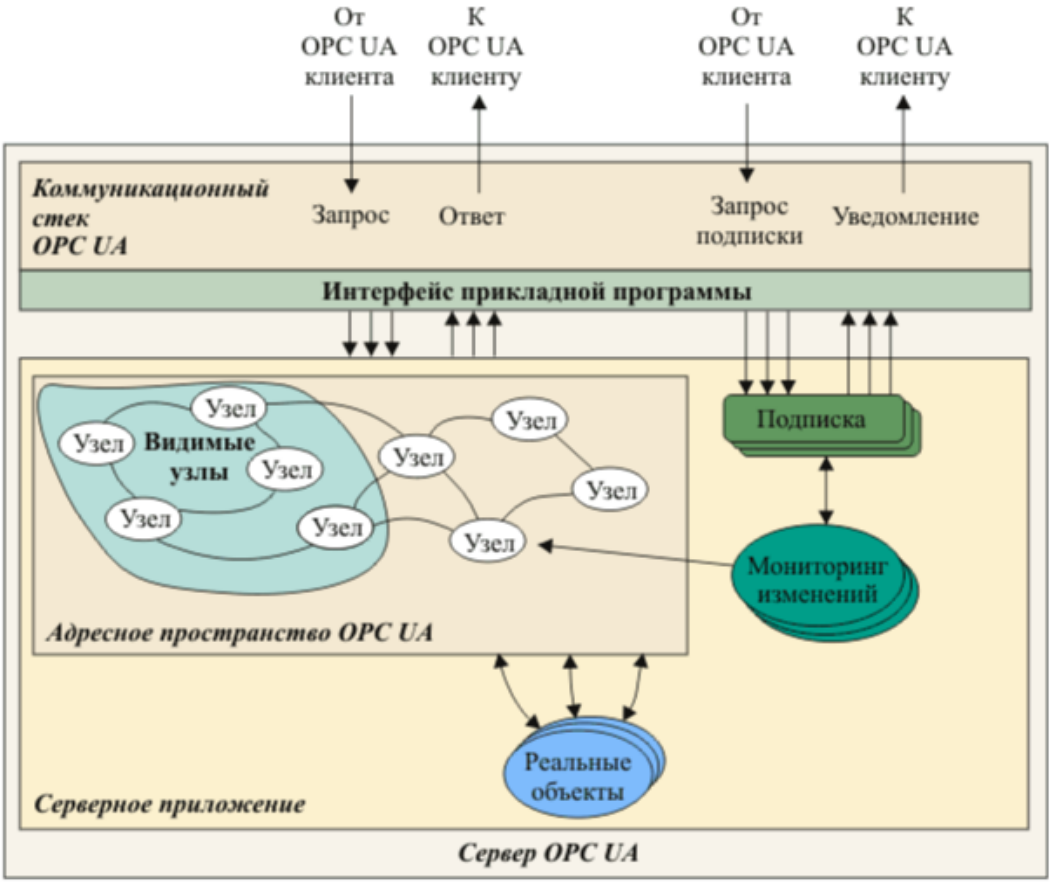
OPCUA спроектований як рішення для міграції з OPC клієнтів і серверів, що базуються на Microsoft COM технологіях. OPC COM сервера (DA, HDA і A&E) можуть бути легко відображені в OPCUA.

Виробники мають змогу самостійно здійснювати подібні міграції або ж рекомендувати своїм користувачам використовувати обгортки і конвертори між цими протоколами. OPCUA уніфікує попередні моделі в єдиному адресному просторі з єдиним набором сервісів.

Підсумовуючи, наведемо короткий опис нових і дуже важливих можливостей, що надає специфікація OPC UA:

1) Підтримка резервування.
2) Пульс з'єднання в обох напрямках. Це означає, що і сервер, і клієнт розпізнають роз'єднання.
3) Буферизація даних і підтвердження їх передачі. Втрата з'єднання більше не призводить до втрати даних. Втрачені блоки інформації доставляються повторно.

Структуру типового OPCUA сервера представлено на мал.5.1. Модулі вводу виведення, ПЛК, інтелектуальні пристрої і програми, які можуть передавати дані через OPC сервер, позначені на ньому як "реальні об'єкти ".



Мал.4.1. Структура OPC UA сервера

Адресний простір сервера являє собою безліч вузлів, доступних клієнтській програмі за допомогою сервісів OPCUA. Вузли в адресному просторі використовуються, щоб представити реальні об'єкти, їх визначення і перехресні посилання. В адресному просторі виділяється підпростір вузлів, які сервер робить "видимими" для клієнта. Видимі вузли організовуються у вигляді ієрархічної структури, для зручності навігації їх клієнтською програмою.

Обмін даними між клієнтом і сервером може виконуватися як шляхом отримання миттєвих відповідей на запити, так і за схемою "видавець - підписник". У другому випадку клієнтська програма здійснює "підписку" на отримання певних даних, які сервер повинен буде надати в міру їх появи. Для реалізації режиму підписки сервер здійснює безперервний контроль вузлів і відповідних їм реальних об'єктів з метою виявлення змін. При виявленні змін даних, виникненні подій або

аварійних сигналів сервер генерує повідомлення, яке передається клієнту по каналу підписки.

4.2 Ідентифікація статичних характеристик об’єкту

Ідентифікацією називають процедуру побудови моделі досліджуваного об’єкту, що проводиться на підставі використання певних методів обробки експериментальних даних, і враховує результати перевірки цієї моделі на адекватність (точність) об’єкту (експериментальним даним) по заздалегідь обраному критерію [6].

Отримана модель певною мірою описує досліджуваний об’єкт і встановлює властиві йому причинно-наслідкові зв’язки. Для фахівців з автоматизації усталеною формою моделей є математичні вирази у вигляді функціональних перетворювачів (операторів), які дозволяють із використанням відповідних теоретичних розділів математики надати досліднику інформацію у найбільш сконцентрованому вигляді. Треба чітко усвідомлювати, що такі математичні моделі (абстракції) ніколи не зможуть гарантувати повного співпадіння з реальним об’єктом, але можуть забезпечити необхідну адекватність об’єкту і бути корисними у вирішенні конкретних завдань з автоматизації, для яких вони розробляються.

При експериментальній ідентифікації дослідник має справу з множиною вимірів, які мають назву статистичної сукупності або обсяг вибірки. На практиці обсяг вибірки завжди обмежений, тому результати обробки експериментальних даних характеризуються випадковістю. При збільшенні вибірки, коли вступає в силу закон великих чисел, результати стають більш точними. Методи збору, обробки та аналізу експериментальних даних є предметом математичної статистики [7].

Часто на початкових стадіях ідентифікації з прагматичних міркувань задля цілей управління об’єкт управління розглядають як «чорну скриню». Принцип «чорної скрині» дозволяє, нехтуючи описом фізичних процесів всередині об’єкту дослідження, на підставі тривалих спостережень (дослідів) отримати адекватну

формальну математичну модель, яка встановлює зв'язок між вхідними та вихідними змінними. Для побудови формальних математичних моделей статики, зазвичай, використовують функціональні ряди [2,3,13] : ряд Тейлора, ортогональні поліноми Чебишева, Лагерра, Лежандра та інші [8].

Так, наприклад, часто використовуються наступні форми формальних статичних моделей для лінійного об’єкта виду:

$$\hat{y} = b_0 + \sum_{i=1}^{n} b_i\, x_i \tag{4.1}$$

та для нелінійного об’єкта виду:

$$\hat{y} = b_0 + \sum_{i=1}^{n} b_i\, x_i + \sum_{\substack{i,j=1 \\ i \neq j}}^{n} b_{ij}\, x_i x_j + \sum_{i=1}^{n} b_{ii}\, x_i^2 \tag{4.2}$$

де $\hat{y}$ - вихідна змінна,

x_i - вхідна змінна

b_i, b_{ij}, b_{ii} - оцінки параметрів рівнянь регресії (4.1) та (4.2).

Експериментальна інформація може бути отримана в режимі пасивного спостереження (без втручання в хід технологічного процесу, роботу устаткування) і в активному режимі за заздалегідь підготовленому плану експерименту з обраними значеннями вхідних змінних.

Оскільки, дуже часто власники виробництв обмежують можливість будь-якого втручання в хід свого технологічного процесу, то при розробці додатку було прийнято рішення зосередитись саме на пасивному спостереженні за необхідними технологічними параметрами роботи системи. Тому, базова реалізація розроблюваної нами системи моніторингу стану пароводяного тракту прямоточного котла працює на основі алгоритму методу Брандона.

Успішний реліз та попит на програму дозволить в майбутньому розширити її можливості й список алгоритмів обрахунку статичних моделей, які вона застосовує.

4.3 Алгоритм методу Брандона

Даний метод призначений для отримання моделей виду:

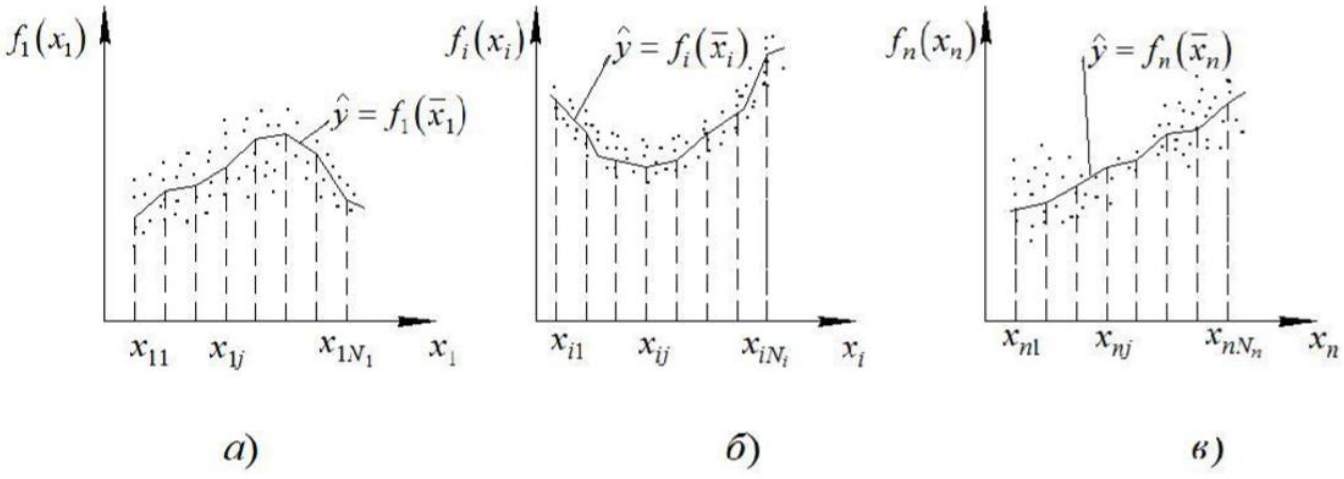
$$y(x_1, x_2, \dots, x_n) = b_0 \sum_{i=1}^{n} f_i(x_i) \tag{4.3}$$

або

$$y(x_1, x_2, \dots, x_n) = b_0 \prod_{i=1}^{n} f_i(x_i) \tag{4.4}$$

де $f_i(x_i)$ – довільна одномірна функція x_i, n – кількість входів, $i = \overline{1,n}$

На точність моделей (4.3), (4.4) впливає порядок розташування вхідних змінних x_i та відповідних $f_i(x_i)$. Для визначення номера індексу кожну $f_i(x_i)$ викреслюють, як функцію однієї змінної і отримують емпіричні лінії регресії (мал. 4.2)



Мал. 4.2. Можливі види емпіричних ліній регресії

Отримавши набір графіків, апроксимацію починають з тієї змінної, для якої розбіжності між експериментальними даними і *y* мінімальні. (мал.4.2.б). Крім того, за формою емпіричної лінії регресії обирають структуру $f_1(x_1)$.

Розглянемо алгоритм методу:

1) За експериментальними даними (обсягом N) викреслюємо емпіричну лінію регресії $y_{x1} = f_1(x_1)$. По її виду визначаємо структуру $f_1(x_1)$ і далі по методу найменших квадратів (МНК) розраховуємо коефіцієнти цього рівняння.
2) Складаємо вибірку нової фіктивної вихідної змінної

$$y_1 = y - f_1(x_1)$$

Фіктивна змінна y_1 не залежить від x_1, тобто

$$y_1 = b_0 + f_2(x_2) + f_3(x_3) + \ldots + f_n(x_n)$$

3) По новій вибірці будуємо емпіричну лінію регресії

$$y_{x2} = f_2(x_2)$$

4) Розраховуємо її коефіцієнти і знову визначаємо вибірку наступної фіктивної вихідної змінної

$$y_2 = y_1 - f_2(x_2) = y - f_1(x_1) - f_2(x_2)$$

Змінна y_2 не залежіть від двох вхідних змінних x_1 та x_2 і визначається наступним рівнянням регресії

$$y_2 = b_0 + f_3(x_3) + f_4(x_4) + \ldots + f_n(x_n)$$

5) Подальшу процедуру визначення функції продовжують до отримання вибірки

$$y_n = y_{n-1} - f_n(x_n)$$

Остання вибірка не залежіть від усіх вхідних змінних, а визначає коефіцієнт b_0 моделі (4.3)

$$y_n = b_0 = \frac{1}{N}\sum_{i}^{N} y_{ni},$$

де N – об’єм вибірки

При використанні моделі зі структурою (4.4) виключення впливу вхідних змінних на вихід виконують у відповідності з виразом (4.5)

$$y_n = \frac{y_{n-1}}{f_n(x_n)} \qquad (4.5)$$

4.4 Структура веб-додатку

Володіючи необхідною інформацією щодо основного алгоритму роботи програми та знаннями принципів роботи OPCUA сервера, наступним етапом в реалізації проекту було прийняття архітектурних рішень щодо побудови самого додатку.

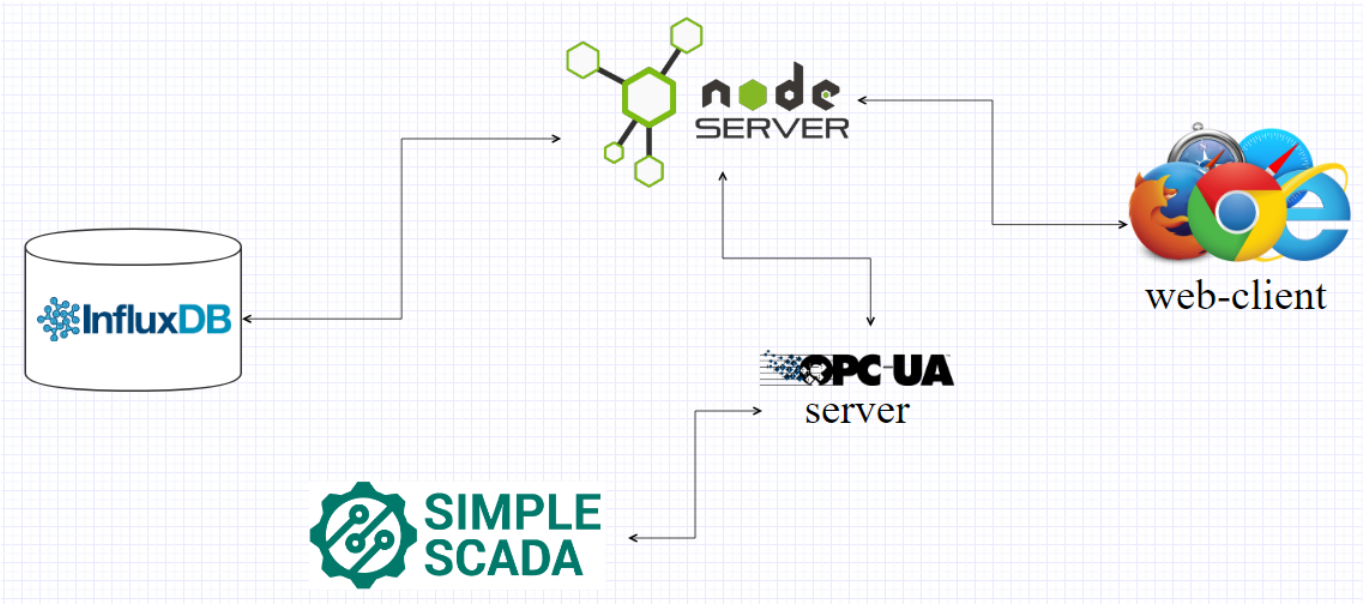
Отримання необхідних користувачу даних можливе за умови якісної та безперебійної роботи двох основних частин програми: серверної (back-end) та клієнтської (front-end). Перша відповідає за збереження використовуваних даних та алгоритми обрахунку статичної моделі об’єкта, а друга – за інтерактивну взаємодію з користувачем.

Оскільки для функціонування додатку не потрібно забезпечувати збереження складно структурованих даних, а основний акцент зміщений в сторону швидкості їх запису (або зчитування), то доцільним є використання бази даних часових рядів,

де інформація зберігається в природньому часовому порядку в попередньо заданому форматі.

4.4.1 Принцип роботи серверної частини додатку

Серверна частина додатку працює на основі доволі популярної та високопродуктивної платформи NodeJs. ЇЇ основним завданням є обмін даними між OPCUA сервером, базою даних часових рядів influxDB та web-клієнтом. Взаємодію цих компонентів можна зобразити у вигляді схеми, зображеної на мал. 4.3.



Мал. 4.3 Схема взаємодії основних компонентів додатку

У верхній частині схеми зображено сервер, кодова реалізація якого виконуватиметься за допомогою платформи NodeJs. Він є головною частиною додатку і пов’язує між собою всі інші компоненти.

Оскільки веб-сервер реалізований в середовищі виконання JavaScript коду NodeJs, то у нашому розпорядженні за замовчуванням перебуває менеджер пакунків npm (Node Package Manager). З його допомогою встановлюємо необхідні програмні модулі, такі як: node-opcua, express, http, influx, bodyParser, numbers та regression.

Функціонал, наданий ними, застосовується на різних етапах роботи додатку і дозволяє в повній мірі реалізувати поставлене завдання.

Розпочинаємо з додавання окремих програмних модулів до проекту. Здійснюється це за допомогою вбудованої функції require(). Фрагмент коду, що відображає цей процес, наведено нижче:

```
const
{AttributeIds,OPCUAClient,TimestampsToReturn,DataType}=require("node-opcua");
const Influx = require('influx');
const express = require('express');
const bodyParser = require('body-parser');
const http = require('http');
const numbers = require('numbers');
const regression = require('regression');
```

Модуль node-opcua являється повноцінною реалізацією стеку OPCUA, що написаний мовою програмування JavaScript. Він дозволяє створювати як OPCUA сервер, так і клієнтів, що з ним будуть взаємодіяти.

Ініціалізація клієнта, встановлення з’єднання з OPCUA сервером та відкриття сесії для роботи з ним, є початковою необхідною умовою для роботи додатку. Аби здійснити підключення до серверу, потрібно вказати його точний URI, що включає в себе ім’я хосту, порт та кінцеву точку OPCUA. В налаштуваннях підключення можна вказати один із режимів безпеки, які різняться між собою наявністю або ж відсутністю цифрових підписів та шифруванням даних.

Найбільш простим і швидким є спосіб анонімного підключенням клієнтської програми до сервера (якщо останній таку можливість надає). Саме цей режим і слід використовувати при розробці програмного забезпечення та тестуванні його працездатності. Перелік необхідних опцій та кодову реалізацію описаних процесів наведено нижче:

```
const options = {
    applicationName: "ClientOpcUa",
    securityMode: MessageSecurityMode.None,
    securityPolicy: SecurityPolicy.None,
```

```
    endpoint_must_exist: false
  };
const endpointUrl = "opc.tcp://DESKTOP-EAMRUI4:4334/UA/MyLittleServer";
const clientOpcUa = OPCUAClient.create(options);

clientOpcUa.on("backoff", (retry) => {
  console.log("Retrying to connect to ", endpointUrl, " attempt ", retry);
});
console.log(" connecting to ", chalk.cyan(endpointUrl));

await clientOpcUa.connect(endpointUrl);
console.log("connected to ", chalk.cyan(endpointUrl));

const sessionForClient = await clientOpcUa.createSession();
console.log("session for client created");
```

Наступним кроком є ініціалізація в JavaScript коді нової (або уже наявної) бази даних числових рядів для подальшої роботи з її функціоналом. Код реалізації наведено нижче:

```
const influxDataBase = new Influx.InfluxDB({
    host: 'localhost',
    database: 'db_Temp_5',
    schema: [
      {
        measurement: 'temperature_1',
        fields: {
          mainValue: Influx.FieldType.FLOAT,
          value_1: Influx.FieldType.FLOAT,
          value_2: Influx.FieldType.FLOAT,
          value_3: Influx.FieldType.FLOAT
        },
        tags: [
          'host'
        ]
      }
    ]
  });
  influxDataBase.dropMeasurement('temperature_1');
  influxDataBase.getDatabaseNames()
    .then(names => {
```

```
    if (!names.includes('db_Temp_5')) {
      return influxDataBase.createDatabase('db_Temp_5')
    }
  })
  .then(() => {
    http.createServer(app).listen(3000, function () {
      console.log('Listening on port 3000')
    })
  }).catch(err => {
    console.error(`Error creating Influx database!`)
  });
```

Обмін повідомленнями з клієнтською частиною додатку проводиться на основі використання протоколу HTPP (Hyper Text Transfer Protocol). Доволі хорошою програмною реалізацію його службових функцій та стабільною роботою з GET і POST запитами володіє бібліотека Express, яка по суті являється швидким, зручним та мінімалістичним веб-фреймворк для додатків побудованих в середовищі виконання Node.js.

Приклад ініціалізації даного модулю, а також кодова реалізація двох публічних API (Application Programming Interface) для отримання усіх наявних OPCUA вузлів та результатів порівняння обрахованої статичної моделі об’єкту з її еталонним видом наведено нижче:

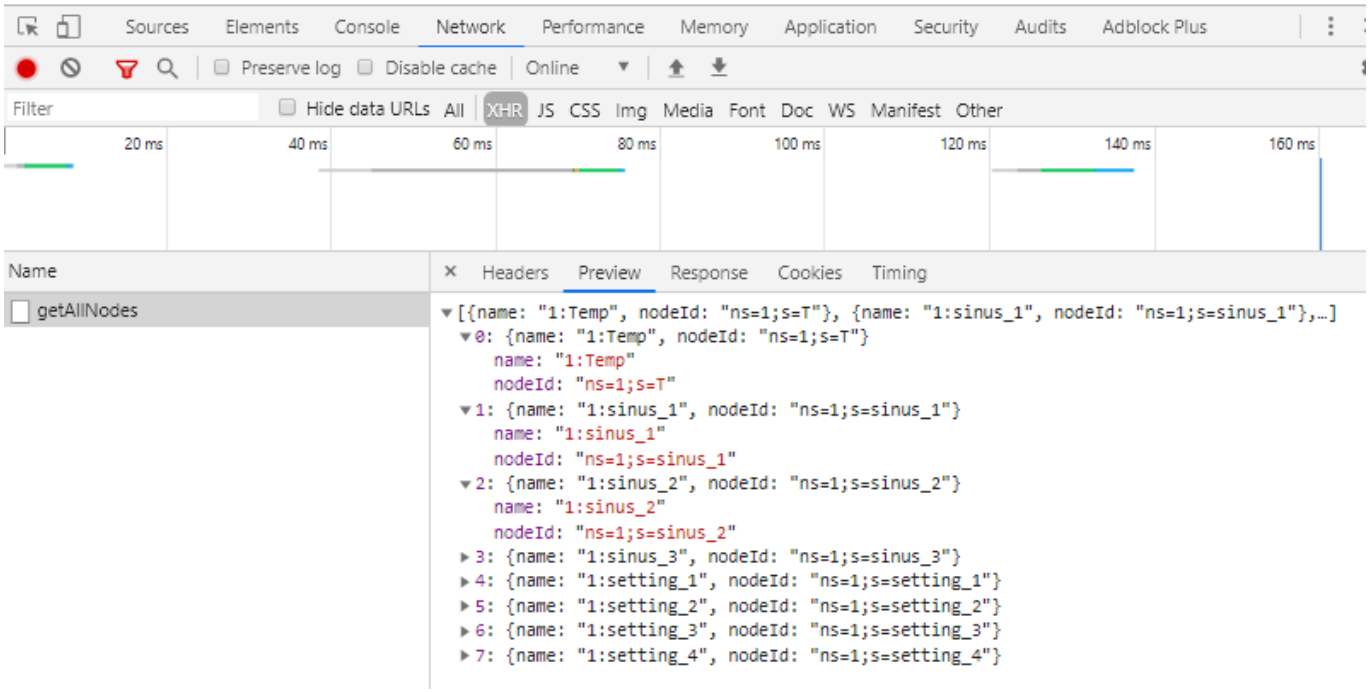
```
const app = express();
app.use(express.static(__dirname + '/'));

app.get('/getAllNodes', async function (req, res) {
  await sessionForClient.browse("ns=1;i=1000", function(err, browseResult) {
    if (!err) {
      let arr = [];
      for (let reference of browseResult.references) {
        arr.push({name: reference.browseName.toString(), nodeId: reference.nodeId.toString()});
      }
      res.send(JSON.stringify(arr));
    } else {
      res.status(500).send(err);
    }
```

```
    });
});

app.post('/getFinallyResult', bodyParser.json(), async function (req, res) {
  initFuncForReadValues(sessionForClient, influxDataBase, req.body);
  res.send(JSON.stringify({res: 'ok'}));
});
```

Коли клієнтська програма надсилає GET запит на адресу '/getAllNodes', то в якості відповіді отримує JSON представлення наявних на OPCUA сервері вузлів і може їх якимсь чином обробити на відобразити для користувача. Приклад такого об’єкту зображено на мал. 4.4.

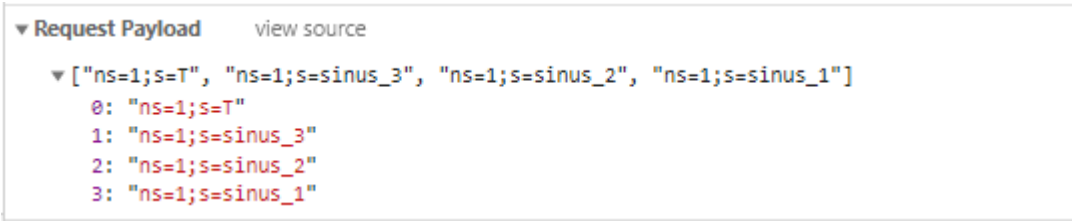


Мал. 4.4 Приклад відповіді клієнту на запит усіх наявних OPCUA вузлів

Серед переліку необхідної для користувача інформації містяться дані про ім’я вузла та його ідентифікатор. Останній необхідно буде повернути серверу при виборі користувачем технологічних параметрів, що підлягають моніторингу.

При здійсненні POST запиту на адресу '/getFinallyResult', клієнтська програма передає дані про основний технологічний параметр, а також масив параметрів які на

нього безпосередньо впливають. Приклад набору даних, що будуть надіслані на сервер, зображено на мал. 4.5.



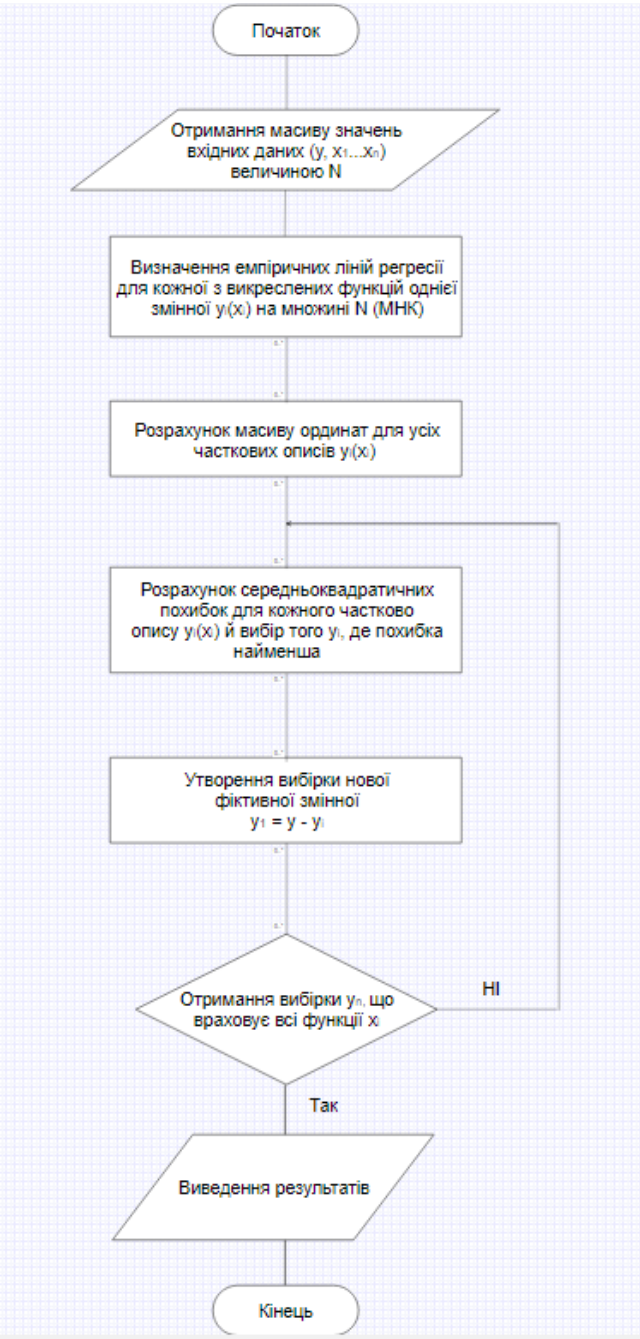
Мал. 4.5 Приклад набору інформації про технологічні пар аметри, що надсилаються на сервер

Функція initFuncForReadValues(), що викликається при отриманні відповідних даних з клієнтської частини додатку, запускає процес зчитування переданих клієнтом параметрів безпосередньо з OPCUA серверу, а також забезпечує їх запис в базу даних і передачу у функцію getResult(), що уже виконує реалізацію алгоритму методу Брандона для визначення статичної моделі об’єкта. Відповідний фрагмент реалізації описаного коду наведено нижче:

```
let timerCount = 0;
  let timer = setInterval(function () {
    timerCount++;
    session.readVariableValue(nodeIdsArray, function (err, dataValues) {
      if (err) {
        console.log(err);
      } else {
        let objArr = dataValues.map((el, index) => {
          if (index === 0) {
            return {mainValue: el.value.value}
          } else {
            let obj = {};
            obj[`value_${index}`] = el.value.value;
            return obj;
          }
        });
        let fields = Object.assign({}, ...objArr);
        influx.writePoints([
          {
            measurement: 'temperature_1',
            fields: fields
```

```
        }
      ]).catch(err => {
        console.error(`Error saving data to InfluxDB! ${err.stack}`)
      })
    }
  });
  if (timerCount === 100) {
    getResult(influx);
    clearInterval(timer);
    timerCount = 0;
  }
}, 1000);
```

Блок-схему, що відображає алгоритм пошуку статичної моделі засобами JavaScript представлено на мал. 4.6.



Мал. 4.6 Структурна блок-схема алгоритму пошуку статичної моделі об’єкта

Фрагмент JavaScript коду, який працює за представленим алгоритмом неведено далі:

```
influx.query(
   `select * from temperature_1
        order by time desc
        limit 100`)
   .then(result => {
     let resWithoutTime = result.map(el => {
       delete el.time;
```

```
  return el;
});

let arrKeyVal = resWithoutTime.map(el => Object.entries(el));

let arr = [[], [[]]];
arrKeyVal.forEach(el => {
  el.forEach((e, index) => {
    if (index === 0) {
      arr[0].push(e[1].toFixed(6));
    } else {
      if (arr[1].length < index) {
        arr[1].push([]);
      }
      arr[1][index-1].push(e[1].toFixed(6));
    }
  })
});

let resultArr = [getNextY(arr[0], arr[1])];

for (let i = 0; i < arr[1].length-1; i++) {
  resultArr.push(getNextY(resultArr[i].next_y, resultArr[i].newArrayX));
}

let sum = numbers.basic.sum(resultArr[resultArr.length-1].next_y);

let a0 = resultArr.reduce((acc,el) => acc + el.cof_A0, 0) + sum / resultArr[resultArr.length-1].next_y.length;
console.log(`a0 = ${a0}`);
resultArr.forEach((el, index) => console.log(`a${index+1}=${el.cof_A}\n arrX${index+1}=${el.removeArr}`));

let a_n = resultArr.map(el => {
  return el.removeArr.map(e => el.cof_A * e);
});

let newToldT_difArr = arr[0].map((el, index) => {
  return Math.abs(el - (a0 + a_n.reduce((acc, current) => acc + current[index], 0)));
});
```

```
    let sq = Math.sqrt(newToldT_difArr.reduce((acc, current) => acc + current*current, 0))/newToldT_difArr.length;
     console.log(sq);
     return sq;
   }).catch(err => {
   console.log(err)
  })
```

Окрім обрахунку середньоквадратичної похибки між поточною та еталонною статичними моделями, це значення також необхідно передати до клієнтської програми і паралельно записати у обраний користувачем вузол OPCUA. Це в свою чергу дозволить налаштувати відображення результатів моніторингу й вивід відповідних попереджень як безпосередньо в додатку, так і в самій SCADA-системі.

Фрагмент коду, що демонструє процес запису даних в OPCUA сервер наведено нижче:

```
const writeVariable = (session, nodeId, value) => {
  let nodesToWrite = [{
    nodeId: nodeId,
    attributeId: AttributeIds.Value,
    indexRange: null,
    value: {
      value: {
        dataType: DataType.Double,
        value: value
      }
    }
  }];
  session.write(nodesToWrite, function (err, statusCode, diagnosticInfo) {
    if (!err) {
      console.log(" write ok");
    }
  });
}
```

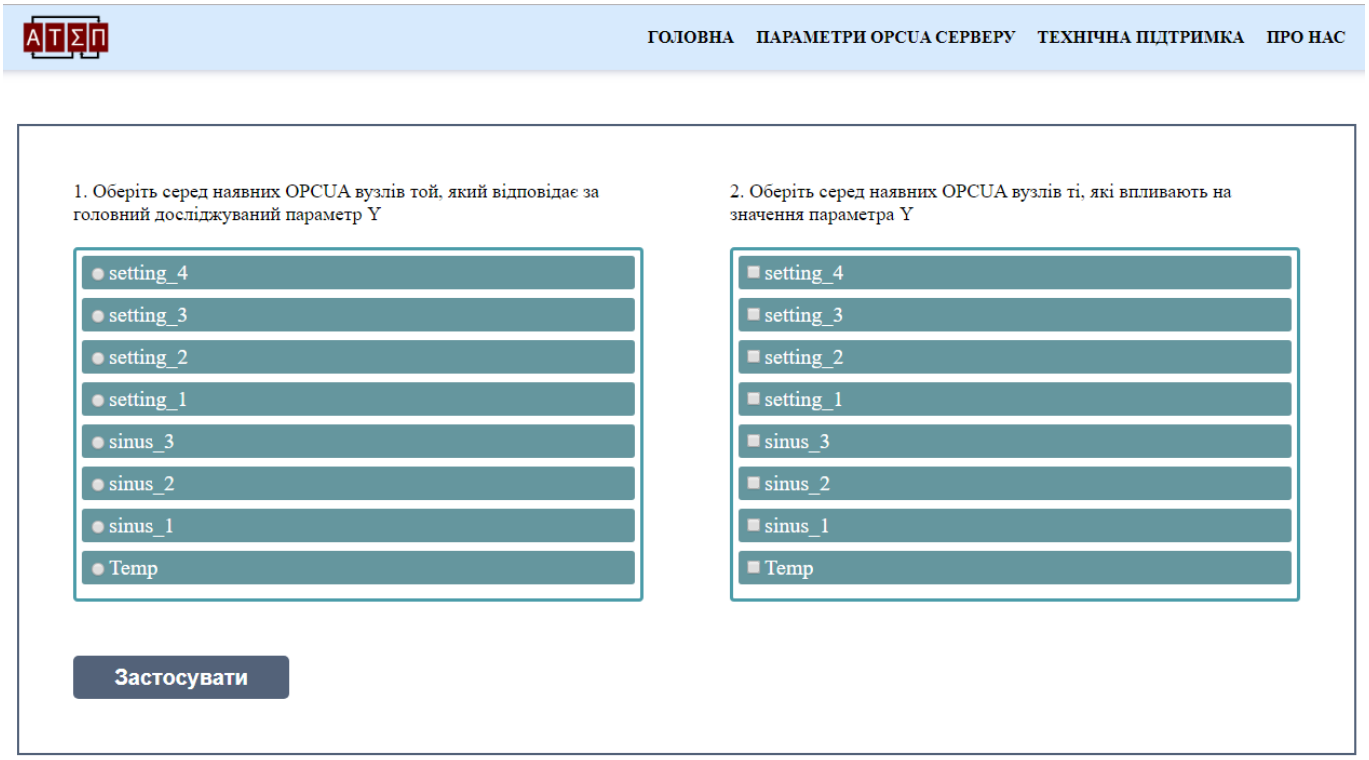
4.4.2 Принцип роботи клієнтської частини додатку

Клієнтська частина додатку в своїй базовій реалізації представляє собою кілька HTML сторінок із власними стилями та JavaScript кодом, що реалізовує обмін повідомленнями із сервером.

Перша зі сторінок дозволяє користувачу ввести певні конфігураційні налаштування. Мова йде про URI (Uniform Resource Identifier) OPCUA серверу, з яким програма повинна працювати, а також ідентифікатор його конкретного вузла, куди будуть записуватись отримані при обрахунках результати. Відповідний скрін даного вікна зображено на мал. 4.7.

Мал. 4.7 Вікно налаштувань OPCUA серверу

У разі переходу користувача на Головну сторінку, йому буде надана можливість обрати необхідні для моніторингу вузли OPCUA серверу, що були попередньо за допомогою AJAX (Asynchronous JavaScript And XML) запиту отримані з серверу, і запустити сам алгоритм програми, натиснувши кнопку «Застосувати». У разі виникнення проблем з даним кроком, користувач зможе перейти за посиланням «Технічна підтримка» та отримати додаткові консультації. Відповідний скрін вікна наведено на мал. 4.8.



Мал. 4.8 Вікно вибору необхідних для моніторингу OPCUA вузлів

Фрагмент JavaScript коду, що реалізовує отримання відповідних даних з серверу, а також надсилання обраних користувачем OPCUA вузлів представлено нижче:

```
document.addEventListener('DOMContentLoaded', function () {
  const $listNodesForMain = document.querySelector('.js-nodes-main');
  const $listNodesForOther = document.querySelector('.js-nodes-other');
  getAllNodesFromApi().then(res => {
    res.forEach((el, index) => {
      $listNodesForMain.insertAdjacentHTML('afterbegin', getNodeItemRadio(el));
      $listNodesForOther.insertAdjacentHTML('afterbegin', getNodeItemCheckbox(el));
    })
  });
  initSendingNodes();
});
const initSendingNodes = () => {
  document.querySelector('.js-send-nodes').addEventListener('click', function () {
    let mainNodeId = document.querySelector('.js-nodes-
main').querySelector('input[name="main-node-radio"]:checked').value;
    let otherModeIds = Array.from(document.querySelector('.js-nodes-
other').querySelectorAll('input[name="main-node-checkbox"]:checked')).map(el => {
```

```
      return el.value;
    });
    sendMainNodeToApi([mainNodeId, ...otherModeIds]);
  });
}
const getAllNodesFromApi = () => {
  return fetch('getAllNodes').then(r => r.json());
}
const sendMainNodeToApi = (arr) => {
  return fetch('/getFinallyResult', {
    method: 'POST',
    mode: 'cors',
    headers: {
      'Content-Type': 'application/json',
    },
    body: JSON.stringify(arr)
  }).then(r => r.json()).then(r => console.log(r))
}
const getNodeItemRadio = (item) => {
 return `<label class="nodes__item">
         <input type="radio" name="main-node-radio" class="nodes__input"
value="${item.nodeId}">
         ${item.name.slice(2)}
        </label>`;
}
const getNodeItemCheckbox = (item) => {
  return `<label class="nodes__item">
         <input type="checkbox" name="main-node-checkbox" class="nodes__input"
value="${item.nodeId}">
         ${item.name.slice(2)}
        </label>`;
}
```

4.5 Імітаційне моделювання системи моніторингу пароводяного тракту прямоточного котла

Для комплексної перевірки працездатності створеного програмного додатку, необхідно надати йому доступ до працюючого OPCUA серверу, в якому наявні необхідні нам технологічні параметри (поточна температура первинного пару та

масив певним чином змінюваних даних, що безпосередньо впливають на її значення).

Окрім того, попередньо було прийнято еталонну статичну модель об’єкта, з якою будуть порівнюватися розраховані при роботі алгоритму дані.

В рамках даної роботи, було вирішено використати для варіації вхідних змінних набори типових функцій синуса з різними амплітудами та зсувами фаз. Загальний вид таких функцій демонструє формула 4.6.

$$x_i = x_0 + Asin(t + b) \tag{4.6}$$

де x_0 – початкове значення,

A – амплітуда,

b – фазовий зсув.

Використовуваний в серверній частині додатку модуль "node-opcua" надає функціонал для реалізації мовою програмування JavaScript не тільки OPCUA клієнта, а й повноцінного сервера з усім необхідним функціоналом. Тому, доцільно в середовищі NodeJs розгорнути власний тестовий OPCUA сервер, в якому описати логіку зміни потрібних нам технологічних параметрів.

Для ініціалізації OPCUA серверу необхідно створити новий екземпляр класу OPCUAServer та вказати перелік потрібних конфігураційних параметрів. Відповідний фрагмент коду наведено нижче:

```
const server = new opcua.OPCUAServer({
  port: 4334, // the port of the listening socket of the server
  resourcePath: "/UA/MyLittleServer", // this path will be added to the endpoint resource
name
  buildInfo : {
    productName: "MySampleServer1",
    buildNumber: "7658",
    buildDate: new Date(2019,10,20)
  }
});
const addressSpace = server.engine.addressSpace;
const namespace = addressSpace.getOwnNamespace();
```

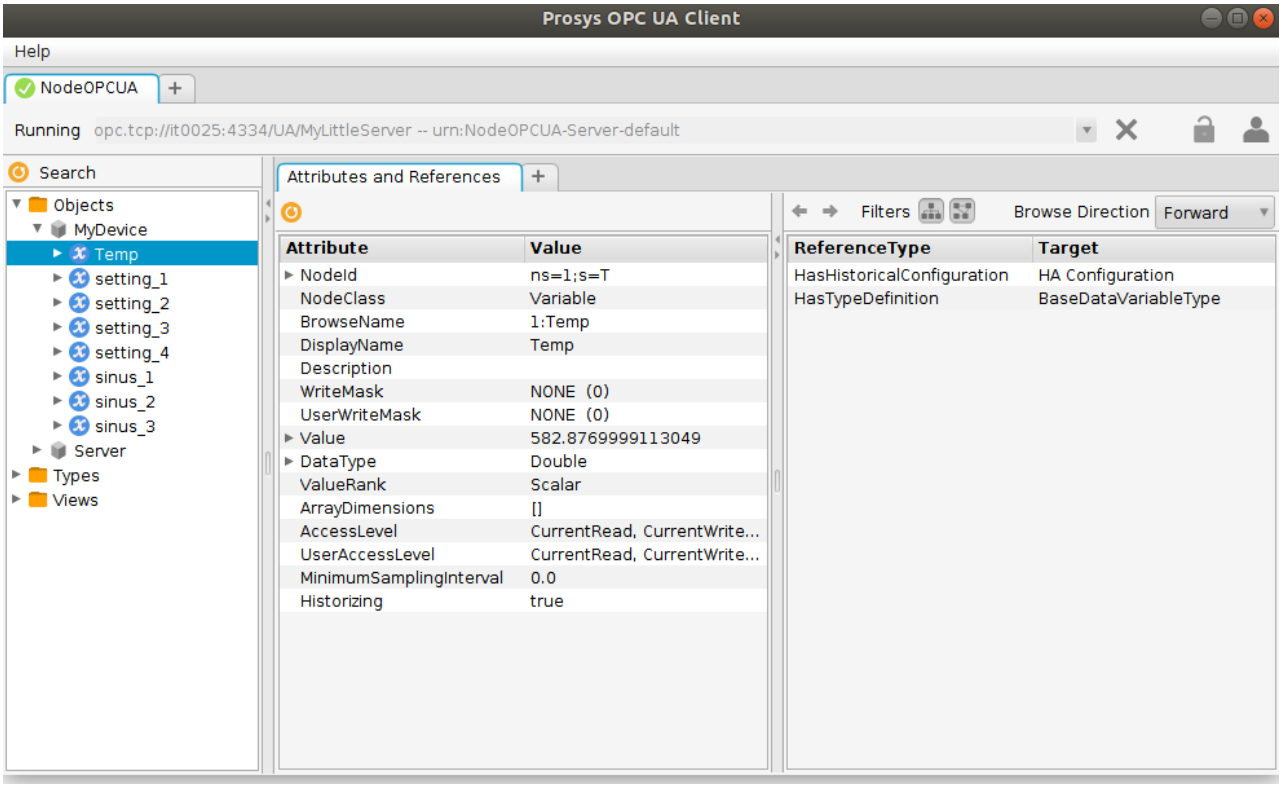
```
const device = namespace.addObject({
  organizedBy: addressSpace.rootFolder.objects,
  browseName: "MyDevice"
});
```

Процес додавання нового OPCUA вузла до серверу є досить простим і забезпечується наявною в модулі "node-opcua" функцією addVariable(). Фрагмент JavaScript коду має наступний вигляд:

```
const TEMPERATURE = namespace.addVariable({
     componentOf: device,
     nodeId: "ns=1;s=T",
     browseName: "Temp",
     dataType: "Double",
     value: {
       get: () => {
         return new opcua.Variant({dataType: opcua.DataType.Double, value: getCurrentTemperature()});
       }
     }
   });
```

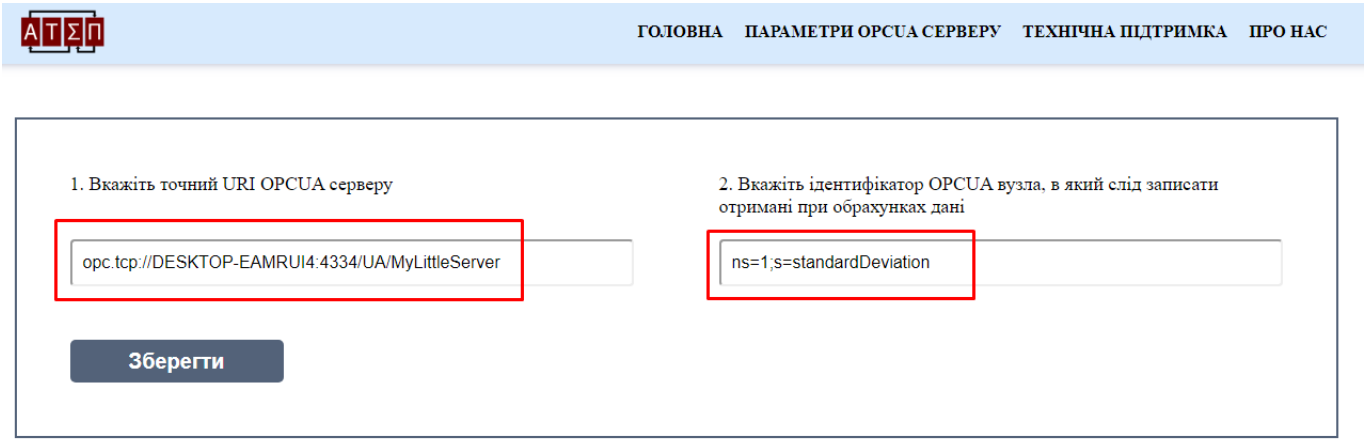
Користуючись цим функціоналом та описавши поведінку функцій синуса, від яких залежить описаний вище параметр "Temp", отримаємо набори потрібних нам для дослідження даних.

Діагностику вузлів створеного серверу зручно проводити програмою Prosys OPC UA Client. На мал. 4.9 зображено вікно даної програми з відкритим списком створених за допомогою JavaScript вузлів. Як бачимо, їхня структура повністю відповідає всім вимогам специфікації OPCUA.



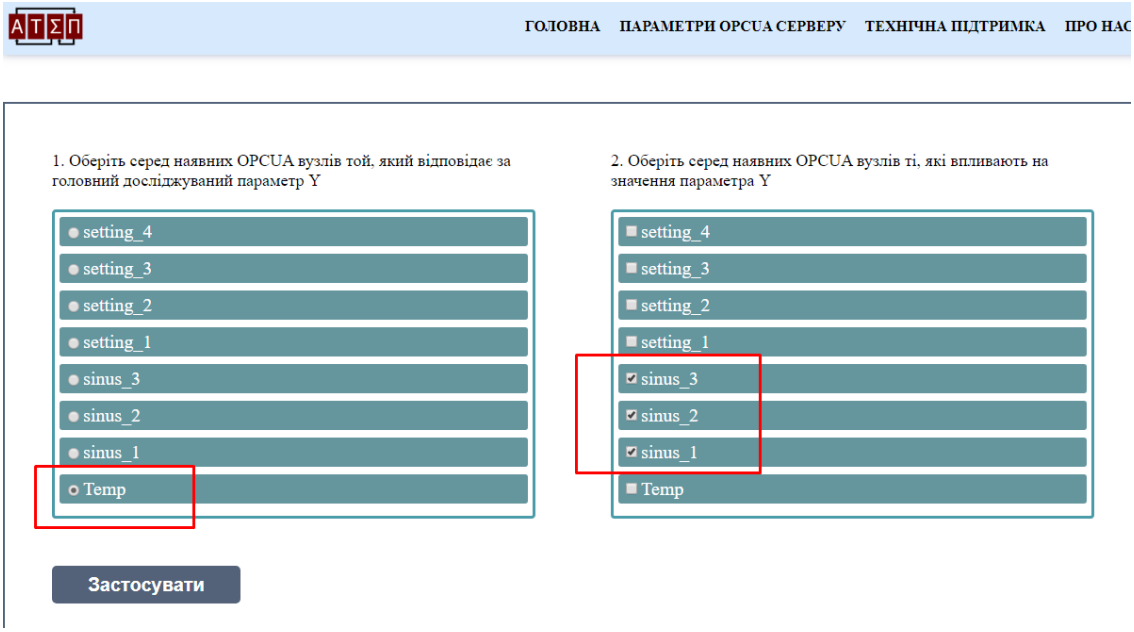
Мал. 4.9 Відображення списку OPCUA вузлів в програмі Prosys OPC UA Client

Тестування функціоналу розробленого додатку розпочинається з вікна налаштувань OPCUA серверу, зв'язок з яким необхідно встановити. В доступних для заповнення полях вказуємо його точний URI та ідентифікатор вузла для запису середньоквадратичного відхилення. Відповідний процес зображено на мал. 4.10.



Мал. 4.10 Введення конфігураційних налаштувань OPCUA серверу

Наступним кроком переходимо на головну сторінку клієнтської частини додатку та здійснюємо вибір основного досліджуваного параметру й списку інших змінних, від яких він залежить. Відповідний процес зображено на мал.4.11.



Мал. 4.11 Вибір необхідних для моніторингу системи параметрів об’єкта

Натиснення кнопки «Застосувати» відправить ідентифікатори обраних OPC UA вузлів на серверну частину додатку, яка на основі даних, що їй приходитимуть з серверу, здійснюватиме обрахунок статичної моделі досліджуваного об’єкту і порівнюватиме її з еталонною. В якості результату виконання в обраний користувачем OPCUA вузол буде записано значення середнього квадратичного відхилення поточної моделі від еталонної.

Задавшись в SCADA - системі відповідним максимально допустимим відхиленням, можна виводити різного роду попередження оператору про зміни поточної статичної моделі об’єкту у порівнянні з початково заданою. Значна різниця між коефіцієнтами даних моделей може свідчити про різного роду фізичні неточності в роботі обладнання, а також буди причиною проведення позапланового обслуговування.

5. РОЗРОБКА СТАРТАП-ПРОЕКТУ

Відносно новим для вітчизняної практики є поняття стартапу, що використовується як спільна назва для радикально нового проекту, який було нещодавно створено, новоствореної компанії або підприємства. Стартап передбачає наявність певної інноваційної бізнес-ідеї, яка наразі не знайшла шляхів своєї реалізації та потребує організаційного оформлення, фінансування, розвитку та ринкової апробації [13].

Стартап – це нещодавно створена компанія (можливо ще не зареєстрована офіційно), що формує свій бізнес на основі інновацій або інноваційних технологій, володіє обмеженою кількістю ресурсів (як людських так і фінансових) і планує виходити на ринок [14].

Стартапи мають певні особливості, які поділяють на дві групи:

1) стартап – це підприємницький проект, який завжди оцінюється вище своєї поточної вартості, а його оцінювання ґрунтується на фінансових прогнозах майбутніх грошових потоків, яким, як правило, надзвичайно складно дати об'єктивну оцінку внаслідок непередбачуваності галузі високих технологій [15];

2) стартап – передбачає наявність компетенцій, достатніх для вирішення різних завдань, зокрема – на початку це технічні завдання, після яких вирішуються операційні. Відсутність компетенцій, необхідних для виведення проекту на наступний етап реалізації стартапу може спричинити нераціонально використання фінансування, що в комплексі виведе проект з ринку. Грошові ж кошти розглядаються як ресурс для застосування компетенцій [15].

Для запуску будь-якого стартапу необхідно зібрати гіпотези бізнес-моделі. Найбільш популярним і актуальним інструментом бізнес-моделювання на сьогоднішній день є шаблон бізнес моделі розроблений Олександром Остервальдером та Івом Піньє. Даний шаблон був представлений в роботі «Построение бизнес-моделей. Настольная книга стратега и новатора» і на даний

момент використовується компаніями різного рівня, від стартапів до транснаціональних корпорацій [12].

5.1 Загальна характеристика ідеї стартап-проекту

Ідея стартап проекту полягає в створенні додатку, здатного в режимі реального часу проводити аналіз статичних моделей технологічних об’єктів керування і робити висновок щодо можливих несправностей в їх роботі.

Сфера застосувань подібних програмних рішень є досить широкою і перспективною. Розробивши базовий необхідний функціонал для роботи додатку, можна уже виходити на ринок і в індивідуальному порядку додавати певну специфіку в програмний продукт залежно від побажань обраного сегменту споживачів.

Основним ринком збуду є підприємства з достатньо високим рівнем автоматизації своїх виробничих процесів, що прагнуть оптимізувати час та вартість планового обслуговування (ремонту) власного обладнання.

Вигоди, які здатен отримати користувач продукту, або ж втрати, яких він може уникнути, можуть коливатись в досить широкому діапазоні. Це може бути як приємним бонусом до щорічних заробітків, так і значною сумою, співставною з вартістю самого підприємства. Погодьтеся, краще вчасно провести технічне обслуговування одного з сегментів виробництва, ніж потім бути змушеним його повністю замінити.

Окрім того, регулярні зупинки в процесі виробництва для діагностики технічного стану наявного обладнання завжди ведуть до певних фінансових втрат. Оптимізація цього процесу і виявлення можливих проблем ще до їх безпосереднього виникнення стане непоганим аргументом, що допоможе підприємству зміцнити як свою безпекову складову так і фінансову.

Підсумовуючи, можна навести список основних переваг, що зможе отримати користувач при роботі з даною програмою:

- Підвищення безпекової складової на підприємстві – більш своєчасне виявлення відхилень в роботі системи і попередження порушень цілісності в її роботі;
- Оптимізація технічного обслуговування – отримання обґрунтованих підстав для позачергового обслуговування або його відтермінування. Це в свою чергу вплине на процес залучення робочого персоналу до таких робіт і фінансових витрат, що будуть при цьому необхідні;
- Зменшення фінансових витрат на виробництві – внаслідок зміни фізичних характеристик елементів автоматизованої системи керування (наприклад пропускної здатності клапанів при утворенні осаду), програмні алгоритми спроектованої системи можуть віддавати команди на управління, які уже не враховують реальний фізичний стан речей, що в свою чергу призводить до збільшення витрат використовуваного для виробництва палива тощо. Своєчасне виявлення цих змін на основі роботи розробленого додатку і усунення їх впливу дозволить оптимізувати використання ресурсів, що безпосередньо вплинуть на фінансову складову усього процесу виробництва.

5.2 Схема прийнятої бізнес-моделі

Таблиця 5.1. Опис бізнес-моделі

КЛЮЧОВІ ПАРТНЕРИ	**КЛЮЧОВІ ВИДИ ДІЯЛЬНОСТІ**	**ЦІННІСНІ ПРОПОЗИЦІЇ**
Основним партнером проекту є кафедра АТЕП КПІ ім. Ігоря Сікорського, на базі якої було розроблено	Проведення досліджень методів ідентифікації статичних моделей промислових об’єктів, розробка серверної та	Підвищення безпекової складової виробництва, оптимізація частоти проведення технічного обслуговування

<table>
<tr><td>частину функціоналу додатку.</td><td>клієнтської частин додатку, технічна підтримка працюючих еземплярів програми.</td><td>обладнання, мінімізація часу простою виробництва – як наслідок ріст доходів.</td></tr>
<tr><td>КЛЮЧОВІ РЕСУРСИ
Ключовими ресурсами є кодова реалізація алгоритмів реалізованого додатку по моніторингу технологічного стану обладнання, а також опис принципів його підтримки та тестування.</td><td colspan="2">СТРУКТУРА ВИТРАТ
Основною складовою витрат є розробка програмного забезпечення та його підтримка, також також важливим пунктом витрат є маркетингове просування розробленої технології</td></tr>
<tr><td colspan="2">ВЗАЄМОВІДНОСИНИ З КЛІЄНТАМИ
Взаємовідносини з клієнтами проводяться на рівні договорів покупки/продажу відповідного програмного забезпечення та зобов’язань щодо його обслуговування</td><td rowspan="2">СПОЖИВЧІ СЕГМЕНТИ
Основним покупцями даного додатку є промислові підприємства з достатньо високим рівнем автоматизації своїх виробничих процесів, що прагнуть оптимізувати час та вартість планового обслуговування (ремонту) власного обладнання.</td></tr>
<tr><td colspan="2">КАНАЛИ ЗБУТУ
Різного роду технічні виставки і конференції, публікація оглядових статтей про роботу системи в електронних засобах масової інформації, соцмережах, email-оповіщення цільових груп клієнтів.</td></tr>
</table>

ПОТОКИ НАДХОДЖЕННЯ ДОХОДІВ

Основним потоком надходження доходів є продаж екземплярів розробленого додатку та обслуговування його роботи. Окрім того, наступним кроком отримання прибутку стане впровадження продажу ліцензій на онлайн версії даної програми

5.3 Технологічний аудит ідеї проекту

Таблиця 5.2 Технологічний аудит проекту

№	Ідея проекту	Технології, необхідні для її реалізації	Наявність технологій	Доступність технологій
1.	Програмна реалізація додатку для розрахунку статичної моделі об’єкту	Платформа Node.js, база даних часових рядів InfluxDB, бібліотеки: node-opcua, regression, influx, socket-io	Технології наявні, потрібно розробити програмну реалізацію пошуку статичної моделі об’єкту на основі методу Брандона	Описані технології знаходяться в вільному доступі

2.	Розробка користувацького інтерфейсу додатку	Платформа Node.js, HTML5, CSS3, клієнтський JavaScript, бібліотеки: socket-io	Технології наявні, потрібно розробити користувацькій інтерфейс для вибору потрібних технологічних параметрів і стартових налаштувань	Описані технології знаходяться в вільному доступі

Згідно обраних технологій, основним фактором успішної реалізації проекту являється використання мови програмування JavaScript, а саме платформи Node.js, в якій будуть створені алгоритми обрахунку статичних моделей об’єкту, а також відбуватимуться процеси обміну даними з OPCUA сервером та користувацьким інтерфейсом.

Використання бази даних часових рядів допоможе в зберіганні отриманої з об’єкта інформації, а розробка користувацького інтерфейсу здійснюватиметься згідно останніх тенденцій – на основі стеку веб технологій HTML5, CSS3 та JavaScript.

Всі описані технології знаходяться в вільному (або частково вільному) доступі і можуть бути використані без додаткових на те фінансових затрат (принаймні в базовій реалізації).

5.4 SWOT-аналіз розроблюваного проекту

Таблиця 5.3 SWOT-аналіз розроблюваного проекту

Перелік сильних сторін:	Можливості проекту:
- забезпечення більш ефективного технічного обслуговування наявного на підприємстві обладнання; - підвищення безпекової складової технологічного процесу, за яким ведеться спостереження; - простий та інтуїтивно зрозумілий інтерфейс для роботи з додатком; - для взаємодії з технологічним об’єктом керування не потрібно складних налаштувань, весь обмін даними здійснюється за загальноприйнятою специфікації OPCUA.	- запобігання виникненню аварійних ситуацій на підприємстві; - оптимізація затрат на виробництво, внаслідок швидкого виявлення неочевидних змін в протіканні досліджуваного технологічного процесу.
Перелік слабких сторін:	Загрози проекту:
- обмін даними для побудови статичної моделі об’єкту можливий лише через OPCUA сервер; - необхідність наявності на ПК клієнта сучасних веб-браузерів для взаємодії з додатком; - не підходить для об’єктів з дуже складними статичними моделями їх технологічних процесів, або потребує певних доопрацювань.	- можливість кібернетичних атак; - відсутність попиту споживачів; - вихід на цільовий ринок програм-кокурентів.

5.5 Взаємовідносини зі споживачами та канали збуту

Взаємодія зі споживачами та знаходження власних каналів збуду для розроблюваного продукту вважаються одними з визначальних факторів, що забезпечують успішність всього проекту. Тому, дуже важливо ще на етапі реалізації поставлених планів чітко розуміти специфіку сегменту ринку, в який ми плануємо увійти.

Визначення базової стратегії розвитку проекту, якої слід дотримуватись впродовж реалізації всіх етапів створення продукту – досить серйозна й клопітка задача. Враховуючи особливості розроблюваної програми, вважаю що концентрована робота з одним цільовим споживчим сегментом, вивчення його специфічних потреб та проблем, дасть в нашому випадку набагато кращий результат, ніж якби ми прагнули охопити весь доступний ринок. Саме обрана стратегія спеціалізації дозволить найкращим чином налагодити тісні зв’язки з нашим кінцевим споживачем і зайняти нішу лідера в цій сфері.

Значна гнучкість та різноманіття технологій, що використовуються при розробці нашої програми, а також можливість розширення базової версії продукту під потреби споживача, дозволить охоплювати одразу кілька спеціалізованих сегментів ринку, тим самим нівелюючи втрати у разі невдалої співпраці з одним із них.

В той же час, основним ринком збуду мають стати підприємства з достатньо високим рівнем автоматизації власних виробничих процесів, для яких дуже важливою є мінімізація часу простою своїх виробничих потужностей і, як наслідок, оптимізація частоти обслуговування виробничого обладнання. В таблиці 3 узагальнимо інформацію щодо взаємовідносин зі споживачами на каналами збуду продукту.

Таблиця 5.4 Взаємовідносини зі споживачами та канали збуту

№	Сегмент споживачів	Особливості поведінки	Вимоги споживачів	Канали збуту	Інші аспекти взаємовідносин
1.	Підприємства державної власності	Необхідність перемоги в тендері задля отримання контракту, досить повільний процес впровадження нових технологій на підприємстві	Використання лише ліцензійного програмного забезпечення, сувора відповідність впровадженої системи державним стандартам	Наукові та технологічні виставки, різні тендерні системи, партнерські компанії	Доволі часто висуваються надлишкові та застарілі вимоги до продукту, процес вирішення виникаючих проблем не завжди оптимальний і часто володіє надлишковим бюрократизмом
2.	Підприємства приватної власності	Важливим є швидший час впровадження системи задля попередньої оцінки результату, перспективи	Відповідність впровадженої системи державним стандартам, додаткові специфічні вимоги до	Наукові та технологічні виставки, компанії-партнери, особисті канали комунікацій	Досить важливим є наявність договору з максимально детальним описом необхідних до

		довгострокової співпраці у разі успішної роботи продукту	роботи продукту	зі споживачами	виконанання робіт, окрім того підприємства можуть мати свої власні специфічні особливості

5.6 Визначення ресурсів та обґрунтування проектних витрат

Витрати, необхідні для реалізації проекту повинні бути досить чітко описані та заздалегідь спрогнозовані. Будь-який виробничий чи організаційний процес завжди містить в собі фінансову складову, яка є якщо не ключовою, то одною з вирішальних.

Структурно проектні витрати поділяються на інвестиційні (пов’язані з формуванням основного капіталу) та поточні (витрати звітного періоду, куди включається як оплата праці робітників, так і оренда робочих приміщень чи обладнання).

5.6.1 Визначення ціни

Таблиця 5.5 Визначення ціни [12]

Рівень цін на товари-замінники	Рівень цін на товари-аналоги	Рівень доходів цільової групи споживачів	Верхня та нижня межі встановлення ціни на продукцію	Розрахункова ціна продукції, тис.грн

Високий	-	Середній/Високий	50000-70000	60000

5.6.2 Визначення обсягу виробництва продукції

Таблиця 5.6 Визначення обсягу виробництва продукції

Показник	Значення по роках				
	2018	2019	2020	2021	2022
Кількість одиниць продукції, яку можливо реалізовувати протягом календарного року	2	6	12	16	20
Ціна одиниці продукції (тис. грн.)	50	60	65	70	70
Вартісний показник випущеної за рік продукції (тис. грн.)	100	360	780	1120	1400

5.6.3 Розрахунок загальних початкових інвестиційних витрат

Для швидкого й успішного старту проекту перелік інвестиційних витрат повинен бути добре обдуманим, чітким і реалістичним. Потенційний інвестор завжди очікує гарно описаний та структурований план фінансових потреб з поясненням чому саме так, і для чого це потрібно.

Таблиця 5.7 Розрахунок загальних початкових інвестиційних витрат

№	Назва етапу	Терміни виконання	Обсяги фінансування, грн.

1.	Проведення попередніх досліджень статичних методів отримання моделі об’єкту	2-3 тижні	7000
2.	Аналіз існуючих програмних платформ та технологій для роботи з OPCUA сервером	1-2 тижні	6000
3.	Розробка серверної частини програмного забезпечення продукту, що реалізовуватиме алгоритми обраних статичних методів ідентифікації моделі об’єкту та забезпечуватиме взаємодію з OPCUA сервером	2-3 місяці	120000
4.	Розробка користувацького інтерфейсу для роботи з програмним продуктом	1 місяць	20000
5.	Тестуванням розробленого продукту на підходящих технологічних об’єктах керування, імітаційних стендах тощо	1 місяць	15500
Разом			168,5 тис. грн

5.6.4 Розрахунок виробничих витрат

Виробничі витрати існують впродовж усіх етапів роботи над проектом і є дуже важливим показником його фінансової складової. Вони включають в себе як

зарплати робітників, так і загальногосподарські витрати, як от орендна плата чи оплата комунальних послуг.

Таблиця 5.8 Розрахунок виробничих витрат

№ з/п	Стаття витрат	Сукупні витрати за період, тис. грн.				
		2019	2020	2021	2022	2023
1	Оплата за оренду робочого приміщення	100	100	95	95	90
2	Оплата комунальних послуг	30	25	25	25	22
3	Витрати пов’язані з рекламними агентами та просуванням продукту на ринку	40	35	25	20	20
4.	Витрати на оплату праці	400	360	320	320	310
5.	Оплата різного роду партнерських програм, необхідних для роботи серверів тощо	40	30	20	20	15
Разом:		610	550	495	460	457

5.6.5 Розрахунок загальних витрат на реалізацію проекту по роках

Таблиця 5.9 Розрахунок загальних витрат на реалізацію проекту по роках [12]

Показник	Значення по роках, тис.					Разом, тис.грн
	2019	2020	2021	2022	2023	
Інвестиційні витрати	168,5	-	-	-	-	168,5
Виробничі витрати	610	550	495	460	457	2572

5.7 Формування надходжень

Володіючи інформацією щодо проектних витрат та даними по можливому прибутку від продажів, можна спрогнозувати перспективи по реалізації цього програмного продукту на наступні 5 років і орієнтовний термін його окупності.

Таблиця 5.10 Формування грошового потоку

№	Показник	Значення по роках					Разом
		2019	2020	2021	2022	2023	
1.	Надходження від проекту (виручка від реалізації продукції) (D)	100	360	780	1120	1400	3760
2.	Загальні витрати (I)	778,5	550	495	460	457	2740,5
3.	Грошовий потік (CF = D - I)	-678,5	-190	285	660	943	1019,5
4.	Акумульований грошовий потік (ACF)	-678,5	-868,5	-583,5	76,5	1019,5	-

ВИСНОВКИ

В рамках виконання поставленого завдання, мною було розроблено автоматизовану систему моніторингу стану пароводяного тракту прямоточного котла ТПП-210А.

Розроблений програмний додаток, що здійснює моніторинг, в режимі реального часу за запитом клієнта розраховує статичну модель об’єкт керування та проводить її порівняння з еталонною.

Необхідні для роботи програми дані отримуються шляхом обміну повідомленнями з OPCUA сервером, а отриманий результат відображається як в клієнтській частині додатку, так і відправляється на OPCUA сервер для можливості подальшої роботи з ним SCADA-системи.

Побудова архітектури розробленої системи є веб-орієнтованою і базується на використанні середовища виконання JavaScript коду NodeJs. Такий підхід дає змогу швидкого розгортання проекту в кінцевого споживача, а також можливість розміщення основних алгоритмів роботи на віддалених серверах з реалізацією найближчим часом системи онлайн підписок на надання подібного роду послуг.

Для якісної роботи програмного забезпечення локального рівня автоматизації було використано контролер фірми Siemens SIMATIC S7-400 з необхідними робочими модулями, який уже не перший рік доводить свою надійність та швидкодію, працюючи на різних об’єктах промислової автоматизації.

В якості технологій супервізорного рівня автоматизованої системи керування було використано середовище розробки людино-машинних інтерфейсів Simple-Scada. Такий вибір SCADA-системи зумовлений досить широким колом можливостей, які вона надає, вбудованим функціоналом для роботи з OPC технологіями та приємним набором графічних інструментів.

Також було проведено роботу по розробці стартап проекту для даного додатку, де визначено ряд сильних та слабких сторін програмної реалізації подібних рішень і окреслено ряд можливостей, які він надає своїм клієнтам.

Проведення аналізу ринку потенційних споживачів та можливого попиту на таку продукцію, дає ознаки вважати подібну ідеї перспективною і рентабельною. Окрім того, веб орієнтованість створеного програмного забезпечення й структура його побудови, надають можливість оперативного внесення необхідних для певної цільової аудиторії змін в його роботу, що дозволить охопити широкий сектор потенційного покупця, а отже й забезпечить ріст попиту на такого роду пропозицію.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Батюк С.Г. Збірник функціональних і структурних схем типових промислових САР, Київ НТУУ „КПІ” – 2001 рік. –50 с.
2. Клюев А.С., Глазов Б.В., Дубровский А.Х. Проектирование систем автоматизации технологических процессов. М.: Энергия, 1980 г.
3. Плетнёв Г.П. Автоматическое регулирование и защита теплоэнергетических установок электростанций. – М.: Энергия, 1976 – 424 с.
4. Волошенко А. В. Принципиальные схемы паровых котлов и топливоподач / А. В. Волошенко, В. В. Медведев, И. П. Озерова. – Томськ: Томський політехнічний інститут, 2011. – 100 с. – (ТПУ).
5. Ідентифікація та моделювання технологічних об’єктів і систем керування : навчальний посібник / В. М. Дубовой. – Вінниця : ВНТУ, 2012. – 308 с.
6. Методи ідентифікації статичних характеристик об’єктів керування : навчальний посібник / В.Ф. Мисак – Київ, КПІ, 2010. – 63 с.
7. Г. Крамер. Математические методы статистики. – М.: Мир, 1975. – 648 с
8. Остапенко Ю.О. Ідентифікація та моделювання технологічних об’єктів керування: Підручник для студентів вищих закладів освіти, що навчаються за напрямом «Автоматизація та комп’ютерноінтегровані технології». — К.: Задруга, 1999. — 424 с
9. API documentation for NodeOPCUA – Режим доступу до ресурсу: http://node-opcua.github.io/api_doc/2.0.0/globals.html
10. API documentation for InfluxDB – Режим доступу до ресурсу: https://node-influx.github.io/class/src/index.js~InfluxDB.html
11. Энциклопедия АСУ ТП – Режим доступу до ресурсу: https://www.bookasutp.ru/Chapter9_2_4.aspx
12. Розроблення стартап-проектів [Електронний ресурс] : Методичні рекомендації до виконання розділу магістерських дисертацій для студентів інженерних спеціальностей / За заг. ред. О.А. Гавриша. – Київ : НТУУ «КПІ», 2016. – 28 с.

13. Розробка стартап-проектів: Конспект лекцій [Електронний ресурс] : навч. посіб. для студ. спеціальностей 151 – «Автоматизація та комп’ютерно-інтегровані технології» та 152 – «Метрологія та інформаційно-вимірювальна техніка» / О. А. Гавриш, – Київ : КПІ ім. Ігоря Сікорського, 2019. – 188 с.
14. Стартап. Вікіпедія — вільна енциклопедія: веб-сайт. URL:https://uk.wikipedia.org/wiki/Стартап
15. Наривончик Д. Инвестиции в стартапы — как это работает? Money Never Sleep: веб-сайт. URL: http://money-never-sleep.ru/startup_investment/